Optimizing Network Patching Policy Decisions

Yolanta Beres, Griffin, Jonathan

HP Laboratories
HPL-2009-153

Keyword(s):
network devices, patching, security analytics, decision support, vulnerability management, policy

Abstract:

Patch management of networks is essential to mitigate the risks from the exploitation of vulnerabilities through malware and other attacks. However, by setting the patching policy for network devices, the IT security team often creates burdens for IT operations or disruptions to the business operations. Different patch deployment timelines could be adopted with the aim of reducing this operational cost, but care must be taken not to substantially increase the risk of potential emergency disruption from exploits and attacks. In this paper we explore how the IT security policy choices regarding patching timelines can be made in terms of economically-based decisions, in which the aim is to minimize the expected overall costs to the organization from patching-related activity. We introduce a simple cost function that takes into account costs incurred from disruption caused by planned patching and from expected disruption by an emergency. We apply a system modelling and simulation approach to produce results that show disruptions caused under changing patch deployment timelines, and use the results together with the cost function to identify the optimal patching timelines. The results from this work can be easily applied by IT security policy decision makers to choose the network patching policy that is optimal for their organization and reflects their risk appetite and network emergency tolerance level.

External Posting Date: July 6, 2009 [Fulltext] Approved for External Publication
Internal Posting Date: July 6, 2009 [Fulltext]

Optimizing Network Patching Policy Decisions

Yolanta Beres, Jonathan Griffin

HP Labs, Bristol, UK
{yolanta.beres, jonathan.griffin} @hp.com

Abstract

Patch management of networks is essential to mitigate the risks from the exploitation of vulnerabilities through malware and other attacks. However, by setting the patching policy for network devices, the IT security team often creates burdens for IT operations or disruptions to the business operations. Different patch deployment timelines could be adopted with the aim of reducing this operational cost, but care must be taken not to substantially increase the risk of potential emergency disruption from exploits and attacks. In this paper we explore how the IT security policy choices regarding patching timelines can be made in terms of economically-based decisions, in which the aim is to minimize the expected overall costs to the organization from patching-related activity. We introduce a simple cost function that takes into account costs incurred from disruption caused by planned patching and from expected disruption by an emergency. We apply a system modelling and simulation approach to produce results that show disruptions caused under changing patch deployment timelines, and use the results together with the cost function to identify the optimal patching timelines. The results from this work can be easily applied by IT security policy decision makers to choose the network patching policy that is optimal for their organization and reflects their risk appetite and network emergency tolerance level.

1. Introduction

Security decisions often involve trade-offs: a security policy choice that optimizes time spent by the security team might create burdens (cost) for IT operations or the business; and a decision to spend money defending one risk means reduced resources for another.

One of the main tasks faced by the security operations team is vulnerability and patch management. The reasoning behind applying patches to remove system vulnerabilities and so reduced security risk is well understood. The longer the systems stay unpatched the bigger the risk that a vulnerability may be exploited by malicious attacks or fast spreading malware.

However, applying patches to network devices such as routers and switches, especially on critical infrastructures, usually has many undesirable implications, mainly in terms of business disruptions. First, it requires dedicated staff who allocate specific amounts of time for patching an individual device. Second, patches and especially network level ones are bound to introduce disruptions, at the minimum level by taking the device offline at the time a patch is applied. Furthermore, any time a piece of network equipment is patched, there is a risk of something going wrong: if the patch fails, or results in unexpected interaction with other devices or current configurations, the disruptions caused can have significant impact on the business, which relies on the network infrastructure. For example, Cisco's devices cannot be patched in the same way as Windows servers: usually the whole network device's operating system has to be replaced with a new one. After patching, it is could easily turn out that the existing configuration does not work with the new OS version, or a routing protocol might end up being broken. Thus, deploying patches across all of these systems in a timely manner is not simple. In addition to the time spent for patch assessment and patch testing, the security operations team often faces restrictions on deploying the patches placed by business in terms of allowed system downtime.

So it is easy to see why network support staff are reluctant to rush in to deploy new patches. At the same time, the effects of an attack on unpatched network devices can be devastating, as many modern

businesses would be crippled if they were to lose their network entirely. Even the announcement of a significant new exploit or piece of malware can cause the emergency escalation of patching, causing significantly increased disruption to the business as parts of the network are taken out of action outside of agreed maintenance windows.

When deciding on the appropriate patching policy, the IT security team in an organization needs to consider if the adopted practice for patch deployment is not exposing the organization to an unacceptable risk of potential exploitation, and to explore any improvements that can be made in the process. To properly manage the vulnerability exploit risk, the security operations team has to try to choose the patch management policy or strategy that optimally addresses those two objectives: minimize business disruptions from planned patches and upgrades, and minimize the time and the number of devices that remain exposed due to being unpatched for a long time.

These two objectives present a conflict, however. To reduce the number of planned disruptions due to patching the operations staff would prefer to adopt patching strategies that span longer periods of time, as more time can be spent on thoroughly testing each patch and also due to the fact that several patches may be released by the vendor when waiting longer and so several patches can be batched together and applied at the same time, reducing the level of disruption from patching. However, from a threat perspective the longer policy would increase the exposure of network devices to potential exploits and attacks due to many devices remaining vulnerable for long periods of time.

From what we have observed, beyond examining historical data, security operations staff have few tools to help them understand such trade-offs or test different vulnerability mitigation strategies before putting them in practice. In the past, patching policies have tended to be treated as "compliance" decisions, e.g., an IT team might choose the longest patch deployment time allowed within guidelines set by audit requirements, or the quickest (i.e., lowest risk) policy that there are enough staff to implement.

The purpose of this paper is to provide analysis of different patching policies using a system model and simulation approach that helps decision makers in organizations to better understand the trade-offs. By using this approach, and factoring in the relative costs to the business of planned and unplanned disruptions, we can change IT security policy choices into economically-based decisions, in which the aim is to minimize the expected overall costs to the organization, while taking into account the extra risks incurred, e.g., by delaying patching.

We show, based on the experimental simulations, how changes in various parameters of the model, such as in the rate of exploits in the threat environment or in the longer patch deployment timeline, affect the planned and unplanned disruption levels, and how these results can help choose an optimal policy that minimizes the overall cost of business disruption due to patching, based on an organization's tolerance of disruption and risk. By investigating different network device patching policies, we will calculate the amount of disruption due to patching that a policy would result in and will estimate the associated increase in risk level. Since the threat environment is ever changing, as demonstrated by recent off-schedule Cisco Security Advisory announcing multiple vulnerabilities [5], we will use the simulations to predict the exposure level under different threat conditions.

The predictive modelling approach provides the advantage of enabling IT security professionals naturally to explore, via experimental results, the dependencies among different decisions in the patch management process and the impact of changes in vulnerability and exploit rates. The approach requires first constructing a model of the dynamics of the operational environment, such as the patching process, which are explored empirically using executable models, such as discrete event simulation, to derive probability distributions of potential outcomes. If the behaviour of the model correctly matches the relevant behaviour characteristics of the underlying environment, we are able to draw inferences about the effects of proposed policy changes from experiments and thus provide support for adopting specific policies.

Our paper is organized as follows. Section 2 describes the network patching problem and introduces the cost functions that would be used for choosing the optimal patching timeline. In section 3, we present the model, constructed to capture the patch management process in the network environment. Section 3

also describes how we model the external threat environment. In section 4 we describe results and analysis from a set of simulation experiments based on the constructed model. The analysis shows the changing levels of planned patching and emergency disruptions under different patch deployment timelines, and suggests some optimal timelines that depend on the specific organization's emergency tolerance level. Section 5 describes results from another set of experiments with changing threat environment, and looks at how this affects the optimal timeline choice. In section 6 we discuss the implications of our analysis and some future work. Section 7 reviews related work in this area. Our paper finishes with some final conclusions.

2. Optimal Policy for Patching Network Devices

The cost of applying patches across a network of potentially thousands of routers and switches is usually very high. So that the installation of patches is done in a consistent manner based on available resources and business needs such as allowed device downtime windows, IT organizations develop a centrally enforced network device patching policy that attempts to balance the benefits and inconveniencies of patching vulnerabilities across network devices. This patching policy includes guidelines for time that should be taken to test patches and the ultimate deadline by when patches have to be applied across all vulnerable network devices.

Due to the complexity of patches for network devices, the testing itself often takes up to 90 days, in practice, which includes an assessment of potential patch failure, and might also include waiting time for a second round of patch releases from the vendor. Because of the high risk of failure when deploying patches that are not properly tested, the security team is often reluctant to expedite patch testing.

Once the patches have been tested, patch deployment takes place. It is during this stage that the security team can be more tactical in choosing how long to set the deadlines in the policy for patch deployment across vulnerable systems. Depending on the size of an organization and on the prevalence of the vulnerable devices, the deployment stage up to the time that most systems are patched can take from half a year to a full year, mainly due to the manual nature of installing patches on network devices. Some organizations that have especially tight limits on possible downtime take even longer to patch or choose not to patch network devices at all.

In this paper we examine several patching policies that differ in the time deadlines set for how long patch deployment should take across the network environment. The network administrators then have to comply with these policies, by scheduling time to patch individual devices. In cases when the deadlines are not met, exceptions have to be raised, requiring authorization and approval for an appropriate delay, thus further increasing the amount of work associated with patching.

By setting a patching policy that allows longer patch deployment deadlines the security team is allowing network administrators more time to schedule patching, and so are potentially reducing the number of exceptions that have to be raised. But *one of the main savings in setting the longer timelines for patching is the reduced amount of device downtime due to patching as administrators are able to batch more patches together and apply them in one shot.* This is especially true for patches on network devices, where patches are usually full OS upgrades, and so each new upgrade generally includes several vulnerability fixes from previous upgrades. These time-based policies have to be set with carefully timed intervals so that new patches are released by the vendor within the set time period and can be batched with previous patches or applied as part of the same upgrade. Below we will examine a wide range of patching deadlines and using model-driven simulations calculate how much operational disruption each would cause.

However, we must remember that the main objective of the patching process is to reduce the risk of network devices being attacked and exploited due to the existing vulnerabilities. The delays in patch deployment usually increase the risk of potential attacks on unpatched, vulnerable devices. The cost of

emergency procedures across these devices in case of the emergence of a successful exploit or root kit is much higher than the operational disruption caused by patching. These emergency procedures could encompass expedited patching, deployment of workarounds, or actual attacks. And so the savings in reduced operational disruption achieved with longer patch deployment timelines have to be weighed against the potential increase in emergency disruption.

2.1. Reducing the business disruption caused by patching

To help determine the appropriate timelines for patch deployment, we first need to define the cost function related to the two forms of business disruption: planned disruption due to patching and unplanned disruption due to emergencies.

We will use the following notation:

- c_{patch} is the cost of applying a patch to a device (or upgrading the OS of a device), which for the purpose of this paper is mainly the disruption caused to the business because the device is offline and not usable;
- $c_{emergency}$ is the cost of applying an expedited fix or a workaround to a device in case of exploit or actual breach/attack; this again is mainly the disruption caused to the business because the device is not usable;
- $p_{emergency}(t)\delta t$ for small δt is the probability that an exploit will emerge during the interval $[t,t+\delta t]$, raising the need for an emergency.

Since an individual organization has hundreds or thousands of devices that might be vulnerable to the same vulnerability and require patching, the overall cost of patching is multiplied across the population of devices, and so the overall cost of one patch is the sum of disruption across the population of devices that require application of this patch: $d_{patch} = c_{patch}\, dev$.

We assume that the cost of applying a patch does not fluctuate significantly across different types of network equipment or from one patch to another[1]. The cost of an emergency, however, is incurred only if a breach is imminent due to the emergence of an exploit or the detection of an actual attack, and so the emergency disruption is dependent on the number of vulnerable (unpatched) devices at time of emergency t[2]: $d_{emergency}(t) = c_{emergency}\, dev\_unpatched(t)$.

Since the arrival of an emergency event is modelled by the probability $p_{emergency}(t)\delta t$, we have an expected value of emergency disruption:

$$d_{emergency} = c_{emergency} \int_{-\infty}^{\infty} p_{emergency}(t)\, dev\_unpatched(t)\, dt .$$

Combining the two types of disruption together, the overall disruption caused by vulnerability management through patching is the patching cost plus the expected cost of emergency:

$$D = d_{patch} + d_{emergency} \qquad (1)$$

The cost of disruption from patching or in case of emergency is obviously organization- and patch-specific, but for our analysis we need to make some simplifications. We can assume that the disruption cost per device caused by emergency procedures is α times greater than the disruption caused by applying a patch. This allows us to state that: $c_{emergency} = \alpha\, c_{patch}$. By substituting this into equation (1) we have[3]

$$D = c_{patch}\left(dev + \alpha \int_{-\infty}^{\infty} p_{emergency}(t)\, dev\_unpatched(t)\, dt\right).$$

Since c_{patch} is constant, the overall disruption cost depends mainly on the number of devices requiring a patch and the expected number of devices that remain unpatched at the time of an emergency.

---

1 This is a simplification as patch quality may vary, and some network devices have a more critical role and cause more disruption when offline.

2 This definition of emergency cost is a simplification, as emergency disruption is really a function of the unpatched population. However, in practice most emergency patching of network equipment happens when an exploit is known about, but before an actual attack takes place (attacks are rare). The resulting disruption is similar in kind to that caused by planned patching, but of greater severity, so we believe this is a reasonable simplification.

3 In this formula we are assuming that there's no discounting of future cost (emergency or later patching) versus cost incurred today (immediate patching).

This cost is incurred for each patch or batch of patches released by the vendor, so if, for example, a vendor releases 3 patches in a year that apply across the same population of devices, the cost *D* triples. In one year an organization usually has to apply hundreds of patches across its various systems and applications. For network devices, the number of patches released by vendors is considerable smaller, usually in tens rather in hundreds per year, which is still quite a large number, considering that a typical large organization might have hundreds or thousands of network devices.

As we said before, the security team can reduce the cost of disruption by being tactical about patch deployment timelines and using these timelines with patch batching capabilities to bundle several patches together and apply them in one shot. For example, by bundling two patches together the administrator would cause half as much disruption, as each device has to be touched once instead of twice. The number of patches that can be bundled together is dependent on the vendor's patch release lifecycle, but in order to meet the deadlines set in the patch deployment policies the administrators would usually start applying one patch across the first set of devices, and once another patch is released will batch it with the previous patch and apply them together across the next set of devices.

Looking at our cost functions the aim with the batching of patches is to reduce $d_{patch.}$ for each patch by reducing the number of *devices* that the patch has to be applied to individually. By incorporating the patch batching effect for each patch we can subtract from the total population of devices the population for which this patch was batched with the next or a superseding patch[4]:

$$d_{patch} = c_{patch}\,(dev\text{-}dev_{batched})$$

By choosing the appropriate patch deployment deadlines that correlate with vendor patch release schedules the aim of the security team would be to increase the size of population within $dev_{batched}$, and thus achieve lower overall patch disruption costs. If d_{patch} was the only cost within *D* the biggest reduction of cost would be waiting for as many patches of possible and applying them together[5].

The other cost within *D*, the cost of emergency disruption $d_{emergency}$, however, increases with longer patch deployment timelines, as the population of devices unpatched at the time of an emergency grows. The aim would be to identify the appropriate patch deployment deadline that decreases d_{patch} (the patch has to be applied across minimum number of devices) while not increasing considerably $d_{emergency}$ (minimum number of devices remaining unpatched in case of emergency), and thus *minimizes* the overall disruption *D* caused by patching.

3. Model of Patch Deployment Process

To explore the effect of different patching policies on the cost of disruption, we use a systems modelling and simulation approach. This approach allows us to accurately capture the characteristics and behaviour of the vulnerability disclose-exploit-patch lifecycle [1, 3, 14], including the patch testing and patch deployment stages. We use a systems modelling methodology based on process algebra and queuing theory [12] that has been developed as a powerful computer simulation modelling technique for framing and exploring complex systems and processes. Within this approach the processes, such as patch deployment, are captured stochastically and events that cause changes in the process, such as vulnerability exploit or patch release, as discrete events whose occurrence is specified with probability distributions or as time-based cycles.

Figure 1 shows pictorially the model that was created of the patch management process in the network environment. The main trigger for this process is the release of a patch or a batch of patches by a vendor.

[4] We made a simplification by assuming that the disruption caused when applying the batch of patches is the same as when applying a single patch. If the batch includes many complex patches, this might not be exactly the case. However, it's common within the network context that the next set of network patches is actually a full upgrade that supersedes or includes any previous patches, and so the cost of applying a new upgrade is the same or is increased only by small delta.

[5] Usually patches are released by the vendor until the end of product lifecycle, and so this would result in the best option being to upgrade a device with the next product without applying any of the patches released before that.

Most major vendors have recently adopted regular patch release cycles, and so we decided to include this in our model, but also recognize that off-cycle patches might be released in between.

From Secunia reports [19] we have examined historical data regarding the patch release frequency by the three major network device vendors adopted across large enterprises: Cisco, HP ProCurve, and Juniper. From these three only Cisco has adopted a regular patch release policy with the main patches being released at 6 months intervals [4], in March and September, and some critical patches in between. Many fewer patches are released by HP ProCurve and Juniper, usually only 1 or 2 per year. Since Cisco networking equipment is by far the most prevalent across large organizations, we decided to use the six-monthly cycle as the patch release frequency in our model. We model the arrival of off-cycle patch releases as a Poisson process with an inter-arrival time based on an exponential probability distribution with the mean time of occurrence set at once per year. The exponential probability distribution was chosen here because of the memoryless property it offers.

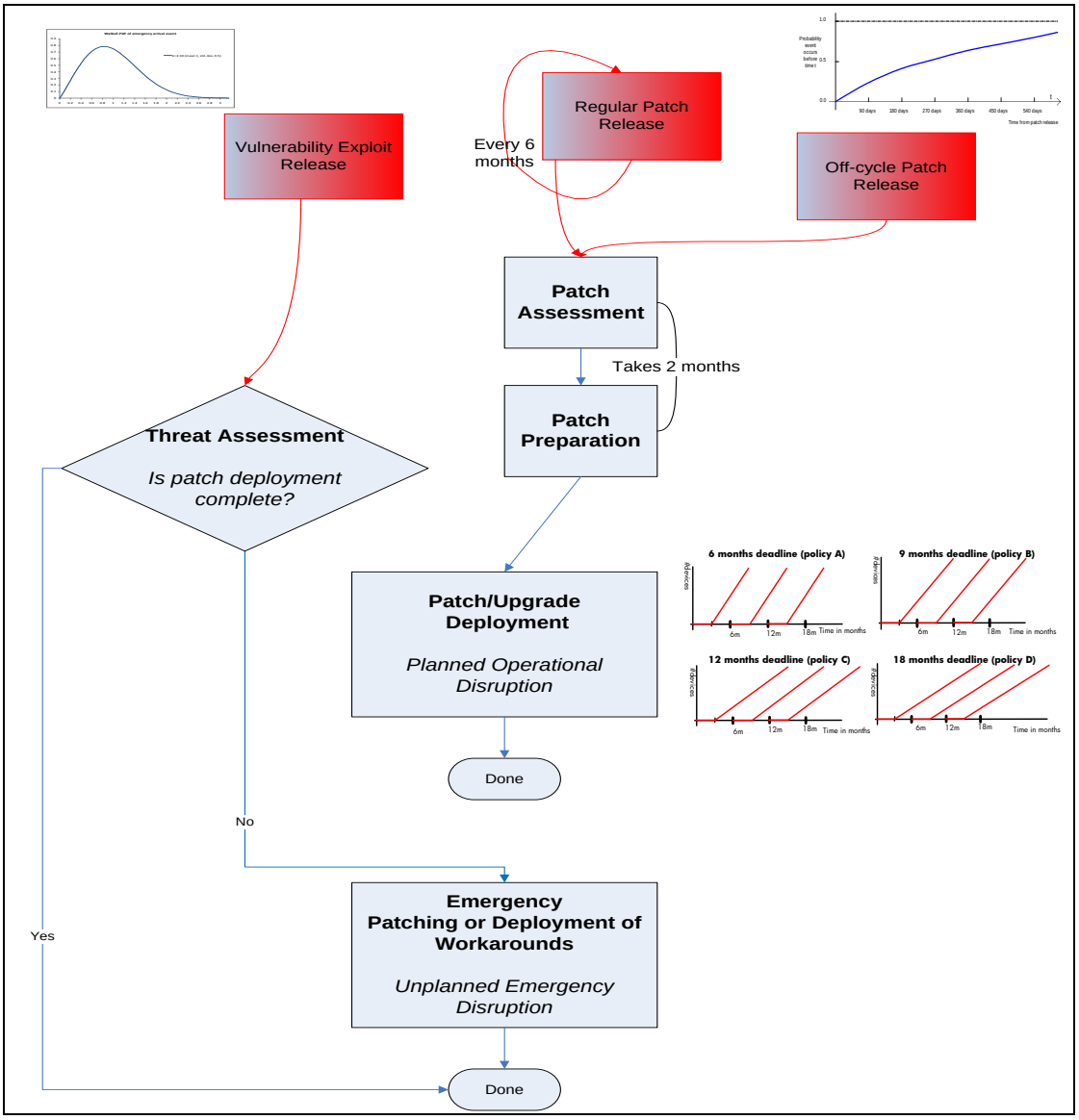


Figure 1. Activity diagram of a typical patch management process.

The process steps taken internally within an organization consist of the previously described stages of patch preparation and deployment. Based on interviews with the network administrators, we decided to have a fixed patch preparation time of 90 days. For the patch deployment stage we needed to determine if during a given time period for patch deployment across a number of devices, these devices are patched individually in regular intervals or in groups. By examining the patch deployment practices in several large organizations, we decided to make some generalizations by assuming that in most cases the network devices would be patched individually at regular intervals so that to meet the deadlines set by the policy. This would result in the linear patch uptake during patch deployment across the vulnerable systems, as is illustrated in graphs on the right hand side within figure 1.

We also assumed that the patch uptake has the same characteristics no matter what length the patching policy is set to; e.g., the devices would be patched at equally spaced intervals when the policy deadline is set at 6 months or at 18 months. Based on the lack of automated tools for patching network devices, and the limited number of allowed downtime periods set by the business, we believe that these two assumptions are not far off the actual network device patching practices.

When we run the experiments with the model, the patch deployment deadlines will be gradually increased with an interval of one month starting with a policy where patches are applied immediately (within one month) and finishing with a deadline for applying patches set to two years.

3.1. Modelling the threat environment

We include threat environment events in the model that cause an emergency when an exploit appears related to the vulnerability being patched. In choosing how to represent the threat environment in our model we have examined previously-announced cases of network exploits. As we have noted before, exploits on network devices are much less frequent compared to the Wintel environment due to the fact that network devices have many different CPU architectures and multiple ranges of platforms, thus preventing effective automatic exploit development. Up to now the attacks and exploits that have been publicly announced have targeted specific versions of architecture and platform, making the likelihood of widespread attacks very low. Within the 4 years 2002-2006 we have found two publicly-announced instances of exploits of Cisco vulnerabilities.[6] Both exploits related to vulnerabilities for which the patch had been released over a year earlier by the vendor. This small number of exploit publications together with anecdotal evidence from the hacker community [7, 8, 9, 11] suggests that exploitation of IOS vulnerabilities is generally difficult, with working exploits taking significant time to be developed after the patch publication by the vendor.

When representing the threat environment within our system model, we concentrated on two factors, the rate of arrival of exploits, and how long after patch publication the exploit appears—which represents the time it takes for attackers/hackers to develop good quality, working exploit code. In the model, the exploit arrival rate is used to determine the likelihood of success in a Bernoulli trial which is conducted after each patch is published, to determine whether an exploit will be developed for this patch. Exploit development time is more complex. It could be represented using mean development time and an exponential distribution, but this doesn't fit the historical or anecdotal evidence. The evidence is better represented by a Weibull distribution, which has 2 parameters, shape and scale, which between them determine the mean and standard deviation of exploit development time.

Therefore, our overall model of the threat environment has 3 parameters: arrival rate of exploits, mean development time of exploit code as measured from patch publication time, and shape parameter of the development time probability distribution. When we come to choose a patch deployment policy, we must estimate these three parameters to determine our model of the prevailing threat environment. Based on the evidence described earlier, our estimate was: 1 emergency (significantly threatening exploit) per year,

[6] March 2004 toolkit exploited vulnerabilities from 2002-2003 [20], Nov 2006 SNMP exploit exploited vulnerability with patch available in 2004-2005 [21].

2 years mean development time from patch publication until exploit, and a shape corresponding to standard deviation of 1 year for exploit development time. The shape of this distribution is plotted in figure 2.

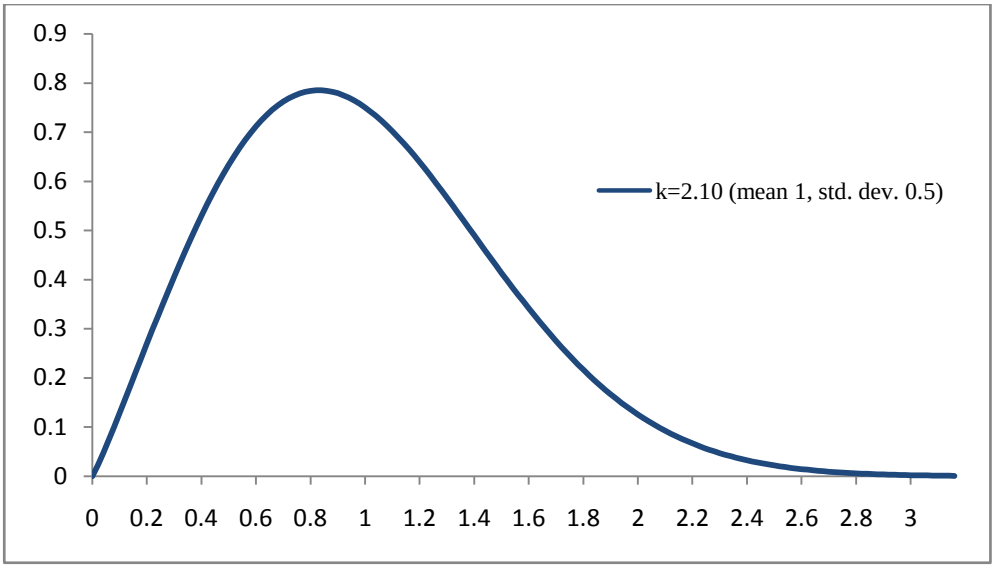


Figure 2. Weibull PDF of an emergency arrival event.

The above parameters will be used in what we call the core simulation experiments, the results of which will be described in section 4. To reflect potential changes in the threat environment such as increased exploit development rate, or in internal patching processes, we will make various changes in the parameters for additional experiments. These will be described in section 5, with further experiments described in the appendices.

3.2. Measuring operational and emergency disruption

During the simulations across different patch deployment schedules, we will be measuring the overall disruption caused by normal patching and emergency procedures. We will be measuring the total disruption caused per year, rather than per individual patch, as this seems as a more practical measure that can be used by the security teams in their policy decisions, since many security policies and budget are determined on yearly basis.

As noted in section 2.2 the disruption from patching mainly depends on the size of the device population that requires a patch to be applied individually. During the simulations, we will measure the relative proportion of the population rather than the exact number of devices, as comparisons across different patching policies will be done based on the relative increase or decrease in disruption with different patching timelines, rather than the exact number. The same approach will be applied for emergency disruption, where we will record the proportion of the population remaining unpatched at the time of an exploit appearing.

4. Results from Simulations with Core Model Settings

In the first set of simulation experiments we look at how disruption changes with increasingly longer patch deployment timelines, with timelines being increased by one month and with the maximum deployment timeline for a patch being 24 months. The result of these simulations is plotted in figure 3. It shows the mean operational disruption from patching per year d_{patch}, and the mean emergency disruption $d_{emergency}$ as longer patch deployment timelines are being adopted by the security operations team.

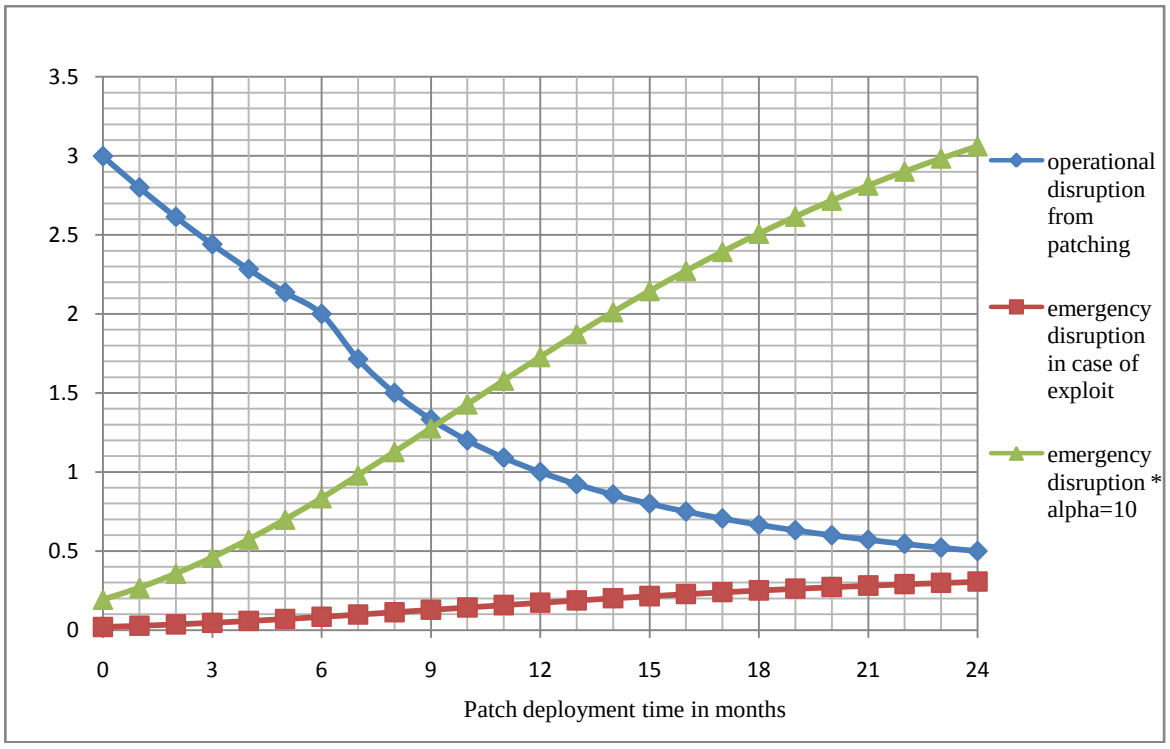


Figure 3. Operational patching disruption and emergency disruption across changing deployment timeline.

As can be seen from these plots, with longer timelines the operational disruption decreases quite substantially, while the increase in the emergency disruption is more gradual and smaller. The savings are even bigger after the 6 month timeline. This point corresponds with the 6 month lifecycle when the vendor releases a new patch. For example, when the patch deployment time is set at 8 months the expected operational disruption is half that when patching policy requires patches to be applied immediately, corresponding to the timeline set at 0 months. However, for policies with timelines longer than 15 months the operational disruption improvements are smaller with each longer timeline.

Since the disruption cost per device caused by emergency procedures is certainly bigger than the operational disruption caused by applying a patch, we decided to scale the emergency disruption by choosing α=10. The green line in the same graph shows this. This time we get a crossover point at a timeline of 9 months where d_{patch}=10 $d_{emergency}$. This represents the patching policy for which the overall disruption is caused by equal measures of operational patching disruption and emergency disruption.

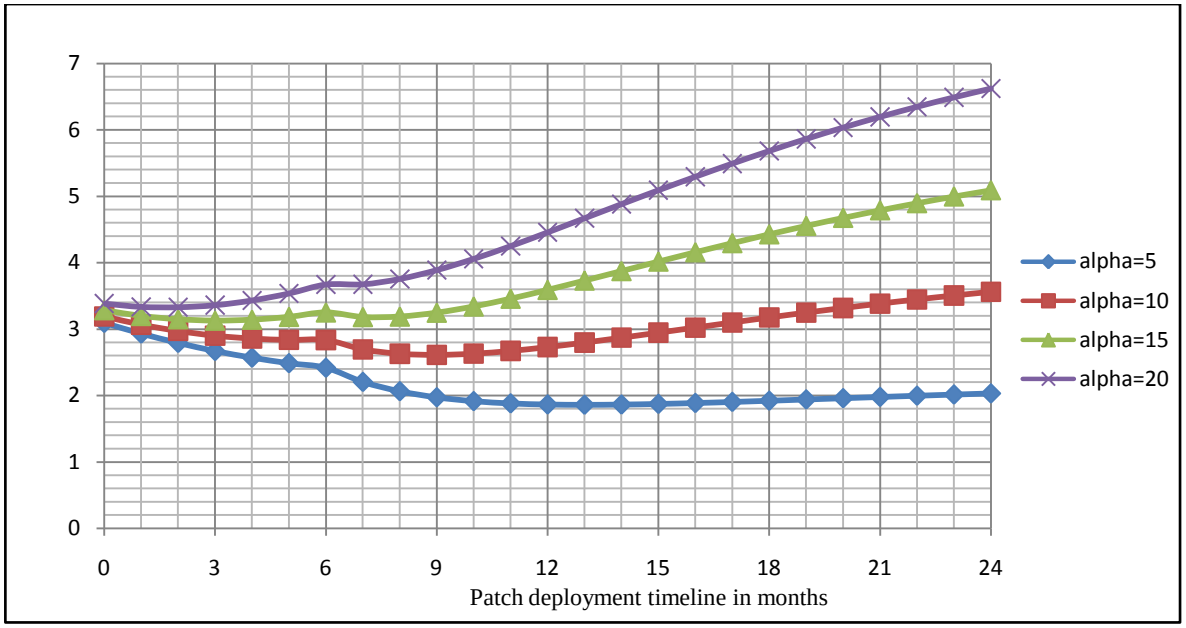


Figure 4. Changes in the overall disruption (operational+emergency) under different values of α.

As we recall from our cost functions described in section 2, the optimal patch deployment deadline would be the one where the overall disruption, $D = d_{patch} + \alpha d_{emergency}$, is minimal for a certain value of α. As the value of α is dependent on a particular organization, its capabilities for dealing with an emergency (such as redundancies across network devices), and its risk appetite, we plotted the overall disruption D under different timelines for several values of α. This can be seen in figure 4.

When α=10 the optimal policy is 9 months and this corresponds to the previous crossover point. If α is larger than that, the optimal policy deadline is much shorter, with α=15 this being at 3 months and with α=20 this being at 2 months. This means that for organizations where emergency disruption is regarded as being more than 10 times worse than the operational disruption caused by normal patch application, the policy should be adopted with patches required to be deployed immediately across vulnerable devices.

If, however, emergency disruption is regarded as similar to or just slightly worse than operational disruption[7], the longer timelines would be more cost effective. For example, when α=5, the lowest overall disruption is achieved when the patch deployment deadline is set at 13 months. But even with timelines longer than that, the overall disruption increases only very slightly.

When we run experiments with even longer timelines, with maximum deployment time corresponding to 5 years, the results of which can be seen plotted in figure 5, a point is reached where the amount of emergency disruption exceeds the operational disruption. This is when the timelines are set to longer than 33 months. It suggests that at the current exploit development level even organizations that do not regard emergency disruption as considerably worse than operational disruption should consider patching their network equipment by 33 months after the patch release date or upgrading it with a newer version, in order to reduce the risk of the network being significantly impacted by an attack or malware.

[7] This is quite likely the case within the current threat environment, where past exploits on network equipment have mostly resulted in denial of service (DoS) attacks, rather than attacks that give a complete access to the router or switch. The DoS attacks are usually not regarded as having big impact due to the redundancies that are commonly built in to the network architecture.

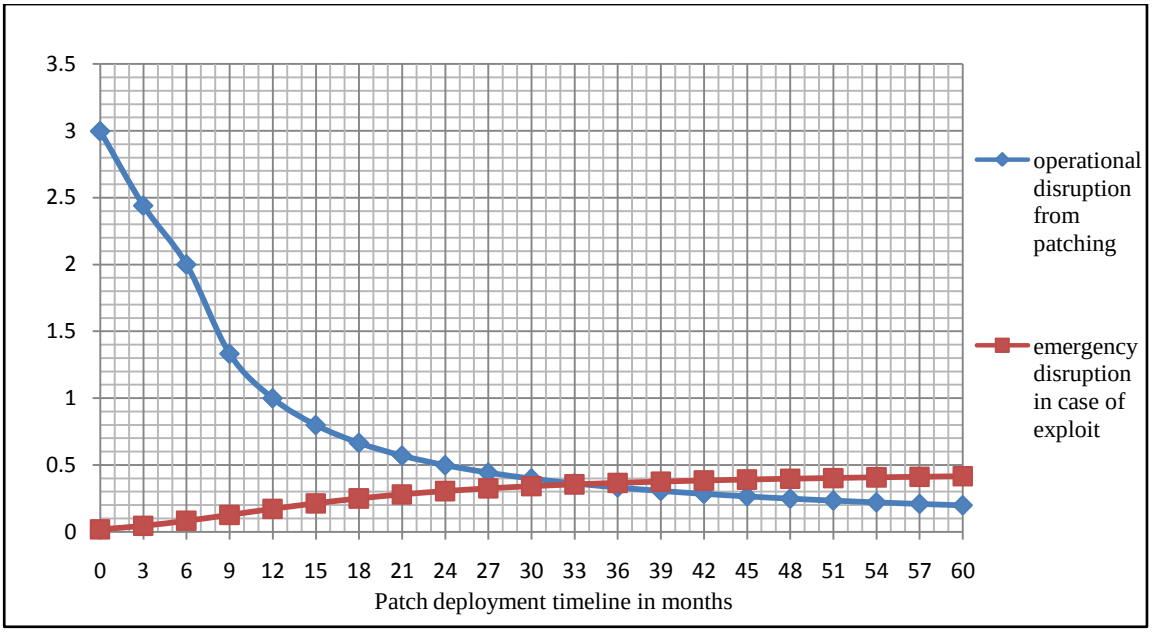


Figure 5. Operational and emergency disruption with long patch deployment timelines.

5. Changing The Threat Environment

The results described in the previous section have been generated from simulations in which the threat environment was described by the Weibull probability distribution as specified in section 3, corresponding to a mean exploit development time after patch publication of 1 year, and an arrival rate of exploits of one every 2 years.

With some of the vendors of network equipment aiming to adopt more uniform OS architectures across their range of network devices, developing exploits that impact network devices might become much easier [10], and so the frequency of exploits may increase and the time for an exploit to be developed may decrease [15].

In the next set of simulation experiments we decided to explore how emergency disruption changes under a worsening threat environment, and how the policy deadlines should be adjusted so that to achieve minimal disruption costs.

5.1. Increased arrival rate of exploits

First we increased the arrival rate of exploits, leaving the exploit development time the same as before set at 1 year. The changes in increased emergency disruption for various emergency arrival rates are plotted in figure 6. As can be seen in the chart, the emergency disruption increases considerably as the arrival rate increases, with the highest increase when an exploit appears every 6 months.

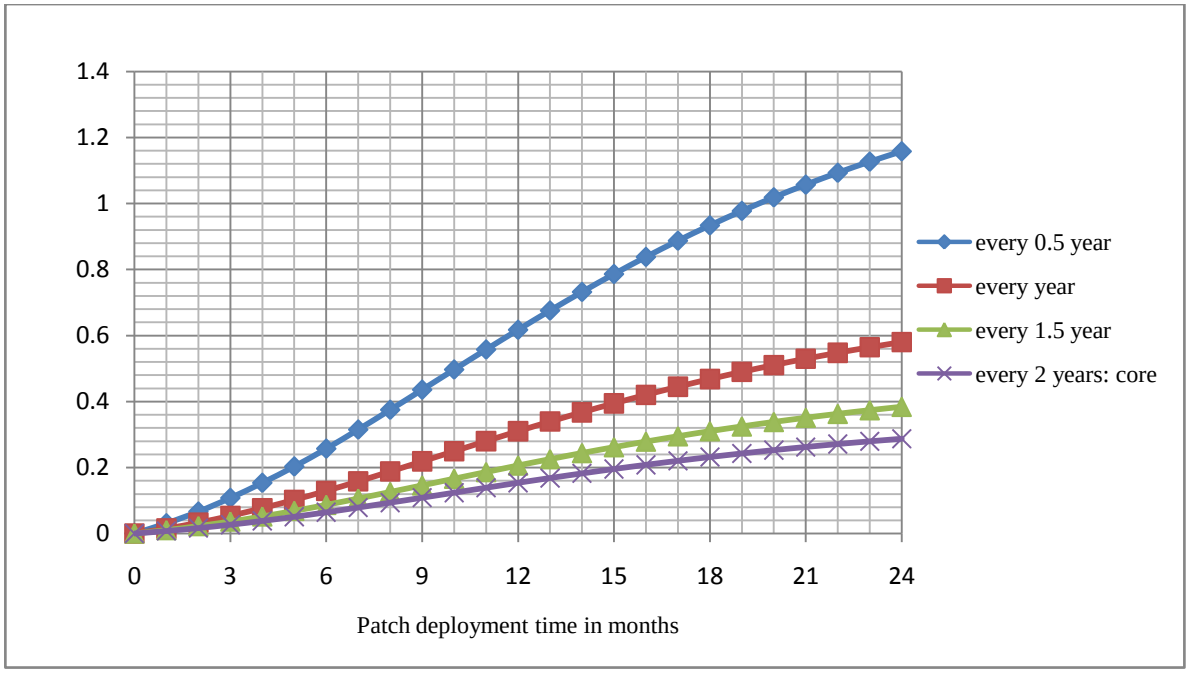


Figure 6. Emergency disruption for different exploit arrival rates, with exploit development time held constant at 1 year.

When we plot the overall disruption with α=10, we can see that with a worsening threat environment the previously optimal policy of 9 months no longer applies. Although with exploit frequency going up from 2 to 1.5 and 1 per year, we don't see a substantial increase in the overall disruption, with the frequency at 2 per year the increase is much bigger. The best option in such a case would be to patch immediately.

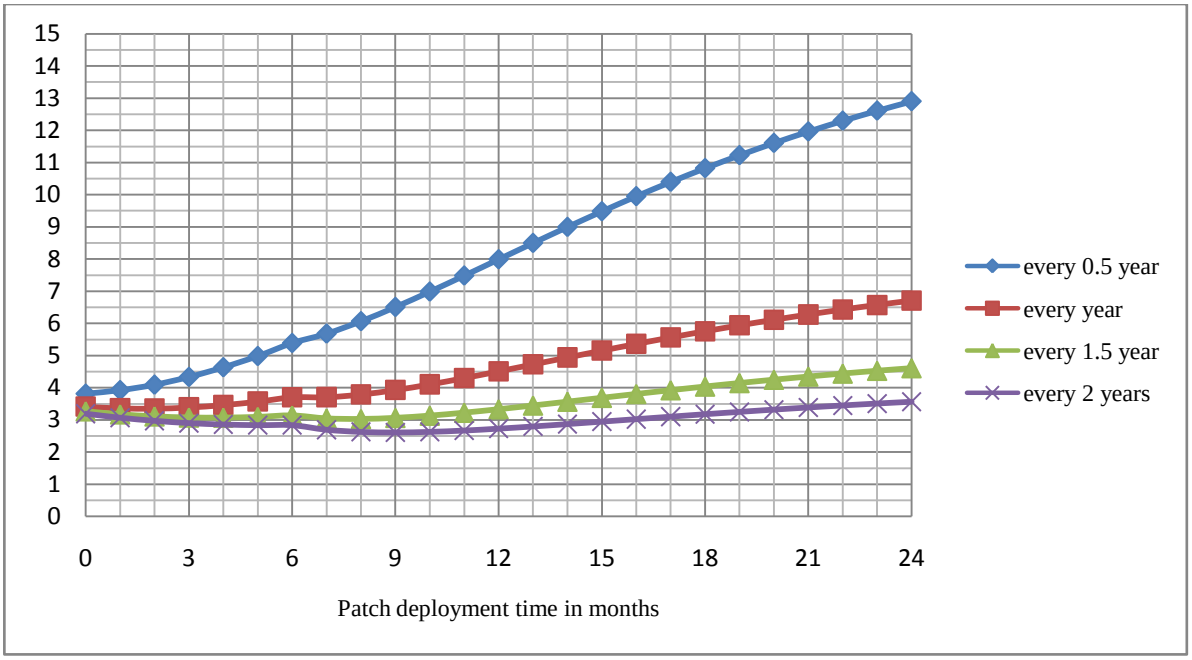


Figure 7. Overall disruption when α=10 and exploit development time is 1 year for different exploit arrival rates.

5.2. Faster exploit development

When we change the mean exploit development time from the initial value of 1 year to 6 months or 3 months, the changes in overall disruption are quite significant, as seen in figure 8. This is particularly noticeable for mean development time of 3 months, which is the same length as patch testing time, so in this case an emergency often occurs before patch deployment has even started. Under such threat environment conditions, both the vendors and the organizations need to re-think the patch release and testing lifecycles and timelines or consider additional mitigation mechanisms.

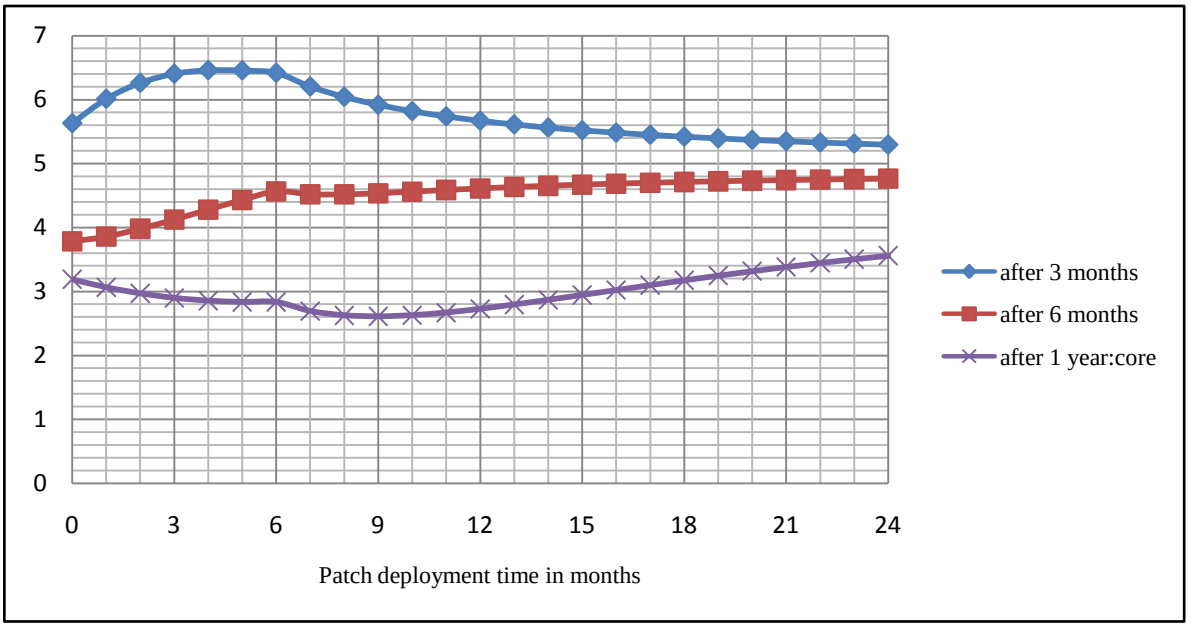


Figure 8. Overall disruption when α=10 and emergency frequency is every 2 years, varying exploit development time after patch publication.

5.3. Increasing exploit frequency under different values of α

Since neither α nor the threat environment is under the control of the security operations team, it is useful to show the impact on our results of changes in both of these parameters. Changes in α may correspond to changes in business conditions, or changes in how the business is run. We represent changes to the threat environment by changes to a single parameter, emergency arrival rate, both to keep the number of axes to two in our analysis, and because increasing emergency arrival rate leads to a smooth and more intuitive increase in the threat level, compared to changes in the exploit development time.

Figure 9 takes as a starting point the patching policy of 9 months, the optimal choice based on the core settings of α=10 and one emergency every 24 months. It shows the difference between the disruption given a 9 month patching policy and that which could be achieved at the optimal policy for a range of values of α and emergency arrival rate. We can see that for small changes in α or a small worsening of the threat environment, there is little difference between disruption at a policy of 9 months and that at the "correct, optimal" policy, i.e., it wouldn't matter too much to the organization if either of these parameters changed a bit or if the estimates of them weren't perfect.

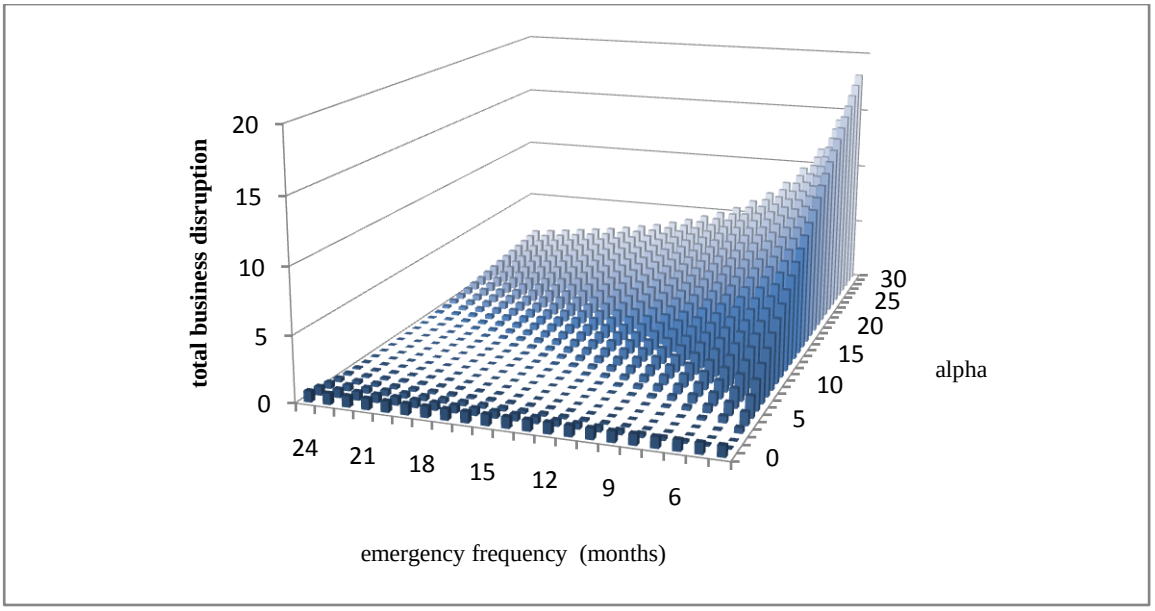


Figure 9. Changes in total disruption at policy of 9 months compared to optimal policy as threat environment and α change.

Figure 10 shows the changing optimal policy under the same varying parameters, and gives a picture of the problem space being explored. In the plot, we can see a number of distinct "regions" of policy: for large α or a high emergency arrival rate, the best policy is to patch as quickly as possible; when α is very small, emergency disruption isn't so important, and the best policy is to patch slowly; then we can see a range of parameter values where the best policy is somewhere in between, with the policy shortening as α increases (more significance for emergency disruption) and the threat environment gets worse (more emergencies).

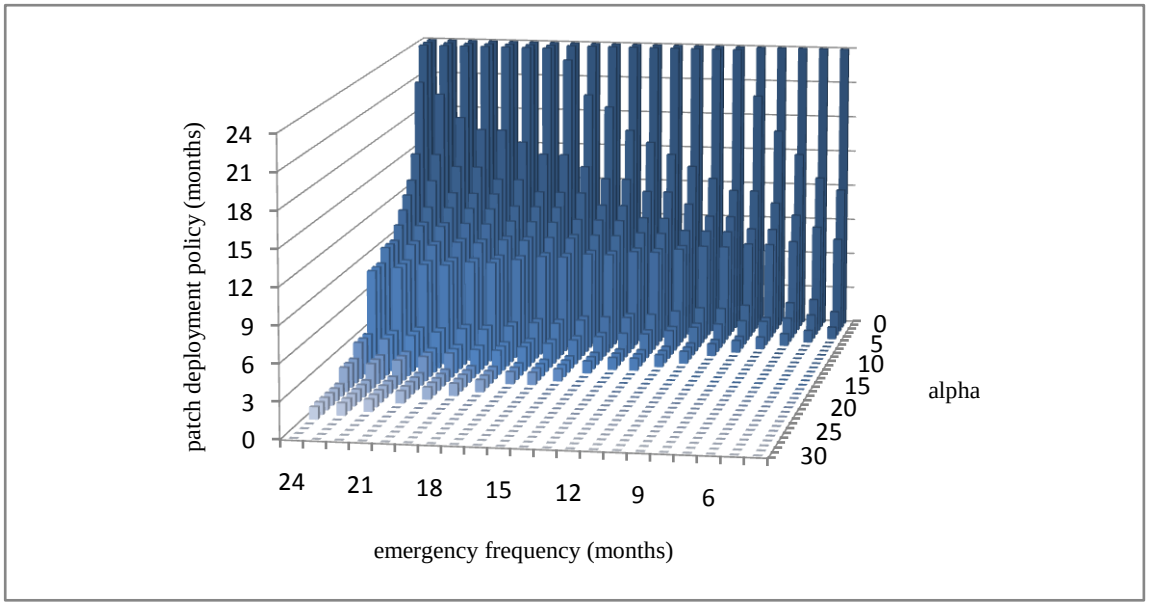


Figure 10. Choosing policy that minimizes total disruption under different threat environment conditions and with varying values of α.

6. Discussion

Our analysis explored the trade-off between operational and emergency disruption by recognizing that normal patching does introduce disruption, which is not negligible, and this disruption can be reduced with longer patching deadlines, that in turn would potentially increase the risk of a security emergency or breach. This analysis was done under certain assumptions about the parameters in the model. The results are most sensitive to the changing threat environment conditions, and that is why we explored the impact of these changes in detail in the previous section. As the threat environment gets worse, with exploits appearing frequently and soon after patch publication, the trade-off between operational and emergency disruption changes, until eventually the only option is to patch immediately. This would be the case for the Wintel environment, and the disruption from patching in such cases is small compared to emergency disruptions from the constant flow of malware and attacks. The reductions in overall disruption in such cases have to be achieved in different way, maybe by implementing faster and less disruptive mitigation approaches. We have explored this in our other work [3]. And so the analysis in this paper is best suited to the context of vulnerability management for the network environment, or other types of environment where exploits and attacks are less frequent (e.g., some server environments).

Sensitivity Analysis

In addition to being dependent on the threat environment, the results from the model could also depend on other assumptions we made, such as linear uptake of patches across the vulnerable population, patch release schedule and so on. The suggestions regarding the optimal policy for patching should be taken in the context of these assumptions. We felt that it is important to do sensitivity analysis, and explore how much results would change under different choices of certain parameters in the model. The results of such sensitivity analysis can be found in Appendixes A to D.

Appendix A explores the effects of using some alternative probability distribution shapes for modelling the exploit arrival time. Appendix B analyses the effects of changing the uptake curve in the model. Currently a linear patch uptake was assumed, which might be too idealistic, and so we ran the experiments with a more realistic 80-20 patch uptake. The results show that although the different patch uptake does cause changes, these changes are quite small. For example, the optimal policy moves from 9 months to 11 months, but the difference in reduced disruption is only at 0.01.

In Appendix C we investigate the effects of "patch coverage"; the proportion of the population of devices that must be patched each time a patch is released. The current assumption is that each released patch would applicable across 100% of population, but with hundreds of different OS architectures for network devices this might not always be the case. The results from these experiments seem to suggest that if the patch coverage is generally low, the reductions in operational disruption from delaying patching will not be as great, and quicker patching policies may be better.

Finally, in Appendix D we explore the effects of different patch release rates. Currently, the 6 monthly patch release rate within the model was based on the patch release lifecycle adopted by Cisco, but we recognize that patch release rate is dependent on vulnerability discovery rate and so may not necessarily be a regular cycle. The results indicate that for deployment policies of less than 6 months, there is difference in increased operational disruption if there are more off-schedule patches. But as soon as deployment policy is >= 6 months, there is no change in operational disruption as any extra off-schedule patches can be batched with one of the other patches.

Cost Function

To help determine the appropriate timelines for patch deployment we chose to minimize the cost function that was defined in terms of disruption. In turn, we decided to simplify the definition of disruption so that it was mainly dependent on the size of the population of devices that would be impacted by the specific task: patching or emergency fix. Both of these can be predicted by running the simulations

on the system model of the patch management process. To apply the cost function within the context of a specific organization the security team would need to choose only one parameter α, which is an estimation of how much worse the emergency fix is to the operations of the business than the planned patching.

This definition of cost function was deliberately kept simple enough, so that it could be easily applied by the security team in various organizations. The more parameters the cost function depends on that cannot be easily predicted or identified based on historical data, the more difficult for the security team to give the correct estimates, and the more reluctant they would be to rely on the analysis and results when choosing the appropriate patching policy.

We recognize that in some cases this cost function might be too simplistic. For example, the extent of emergency disruption might be different depending on the type of vulnerability being exploited, and in such case α should be defined as a function that is dependent on the criticality of exploits or emergencies.

Also our current interpretation of an emergency deals with an expected value that is only dependent on the number of devices unpatched and vulnerable at the time of the emergency. This might be too simple as the threats to networks become more sophisticated, and impact not only network devices, but critical business applications and transactions. So a more complex definition of emergency disruption might need to be developed, that takes into account the organization's risk appetite and ability to tolerate emergencies. This might also require a more complex system model that captures how an organization reacts to an emergency, including the effect of various processes and security mechanisms that are deployed to deal with the emergency.

Dealing with Non-quantifiable Risks

When describing the threat environment in the network space we quantified it using probability distributions, assuming that the likelihood of network exposures can be estimated with some accuracy. However, we acknowledge that until recently widespread network exploits and attacks have occurred too rarely to get a handle on their statistics, and in general these are non-random events because network attacks are usually targeted by some adversary that is causing them to occur at his/her convenience. Such exposures are often referred to "non-quantifiable risks" [18].

Since the probability of such events cannot be accurately taken into account, an important additional factor is estimating the consequences of such events. Impact analysis [17] can help here in understanding if even a rare event can cause severe damage to an organization. To properly manage non-quantifiable risks the overall network security defense measures should not only depend on the network device patching policy, but also on exposure detection and efficient defence and recovery at the time of the actual attack.

7. Related Work

In recent years there has been some work examining the trade-offs involved in different patching policies, but none that specifically address the patching of network devices. Beattie, et al. [2] were the first to explore the factors affecting the best time to apply patches so that organizations minimize disruptions caused by defective patches. Their results indicate that patching during the period of 10 to 30 days after first patch release date is the optimal period for minimizing the disruption caused by defective patches. In our work, rather than looking just at a single patch and adjusting a single point in time to start patch deployment, we analyse the overall patch management process that also takes into account the time taken to apply the patches across all the vulnerable network devices in an organization; this can be considerable for a large organization.

Radianti, et al. [13] explore proactive and reactive patching policies using system dynamics modelling. Although their approach of modelling and simulation is similar to ours, the main difference is in the type

of policies that are chosen to explore. Radianti, et al. explore generic patching policies, whereas we aimed specifically at exploring patching of network devices, as these represent a very special case.

The work by Zhang, Tan and Dey [16] provides an analytical framework for cost-benefit analysis of different patching policies. They assume that patching lead time (the time taken to apply a patch across the systems) is negligible or very small comparing to the overall patching lifecycle, which we argue is not the case in large organizations with thousands of systems requiring the same patch. The authors also assume that the costs associated with vulnerability exploitations can be estimated with relative ease by an organization, which in reality is very difficult to determine with any accuracy. We believe that our proposed simulation-based approach allows an organization more naturally and flexibly to explore the pragmatic outcomes from different patching policies than a purely analytical framework would allow.

Another stream of research relevant to our work here is the analysis of the vulnerability life-cycle, and their past and future trends. The earlier work by McHugh and Arbaugh [1] introduces a life-cycle model for system vulnerabilities. Looking at CERT data on attacks related to specific vulnerabilities, McHugh and Arbaugh noted that many systems were still being attacked months or even years after the patches became available for the vulnerabilities exploited in those attacks. Since then, security operations teams in many organizations have streamlined their patch management processes so that fewer systems remain vulnerable for so long. However, the rate at which new vulnerabilities are discovered and exploited is constantly increasing, thus requiring regular reassessment of existing patching practices. The work by Frei et al [6] is important here for helping understand past and current trends of how vulnerabilities have been exploited and how mitigations have been handled by software vendors. This work looked at the generic set of vulnerabilities as disclosed by CERT, Security Focus, and others. However as we noted before, for network device vulnerabilities the incidents of actual exploits are rare, and so the same probability distributions functions as for generic vulnerabilities cannot be applied. We believe that more analysis would be beneficial to assess the appropriate probability distributions in the network vulnerability space.

8. Conclusions

In this paper we explored how IT security policy choices regarding patching timelines for network equipment can be made in terms of economically-based decisions, in which the aim is to minimize the expected overall costs to the organization from patching-related activity. We introduced a simple cost function that takes into account costs incurred from disruption caused by planned patching and from expected disruption by an emergency when an exploit or malware emerges. By lengthening the required patch deployment timelines, the IT security policy decision makers can reduce the disruption caused by planned patching as more patches can be batched together, but this would increase the expected emergency disruption as the devices would remain unpatched for longer. We applied a system modelling and simulation approach to explore the disruptions caused by changing patch deployment timelines within the range of 0 to 24 months, and used the results together with the cost function to identify the optimal patching timeline. When modelling the network vulnerability management process we tried to capture current network patch management practices used across large organizations, and modelled the network threat environment based on historical (though sparse) data on network exploits over the past 4 years. The resulting optimal patch deployment policy of 9 months should be viewed as optimal under these assumptions. By increasing the frequency of exploits in the next set of experiments we saw that this 9 month timeline soon stops being an optimal policy, with the best option being to patch immediately.

We believe that the results from this work can be easily applied by the IT security policy decision makers in their respective organizations to choose the network equipment patching policy that is optimal for their organization and reflects their risk appetite and network emergency tolerance level. It is our hope that this approach may one day form best practice to follow not just in choosing patching policy but in other areas of security decision-making.

Acknowledgements

We are grateful to Matthew Collinson from HP Labs for his advice on description of the cost functions, to Benedict Addis from HP Labs for the comments and suggestions regarding the the threat environment, and to Peter Ventura, Max Heitmann, and David Markle from CITI for providing valuable insights into the patch management process within a large organization.

References

[1] W.A. Arbaugh, W.L. Fithem, J. McHugh, "Windows of Vulnerability: A Case Study Analysis", *IEEE Computer*, 2000.

[2] S. Beattie, S. Arnold, C. Cowan, P. Wagle, C. Wright, A.Shostack, "Timing the Application of Security Patches for Optimal Uptime", *Proc of LISA'02: 16th System Administration Conference*, 2002.

[3] Y. Beres, J. Griffin, M. Heitman, D. Markle, P. Ventura, "Analysing the Performance of Security Solutions to Reduce Vulnerability Exposure Windows", *Proc. of 2008 Annual Computer Security Applications Conference*, Dec 2008.

[4] Cisco Security Advisories and Notices, "The publication schedule for Cisco Internetwork Operating System (IOS) Security Advisories", March 2006, http://www.cisco.com/en/US/products/products_security_advisories_listing.html

[5] Cisco Security Advisory: Multiple Vulnerabilities in the Cisco ACE Application Control Engine Module and Cisco ACE 4710 Application Control Engine, February 2009, http://www.cisco.com/warp/public/707/cisco-sa-20090225-ace.shtml

[6] S.Frei, M. May, U. Fiedler, B. Plattner, "Large-Scale Vulnerability Analysis", *Proc. of SIGCOMM'06 Workshops*, September 2006.

[7] Felix "FX" Lindner, "Cisco Vulnerabilities", BlackHat Federal 2003, http://www.blackhat.com/presentations/bh-federal-03/bh-fed-03-fx.pdf

[8] Felix "FX" Lindner, "Development in Cisco IOS Forensics", Defcon 11, 2003http://www.defcon.org/images/defcon-16/dc16-presentations/defcon-16-fx.pdf

[9] Technical interview, "Exploiting Cisco with FX", 2005, http://www.securityfocus.com/columnists/351/2

[10] Felix "FX" Lindner, "Cisco IOS-Attack and Defense: State of the Art", 25th Chaos Communication Congress, December 2008, www.phenoelit-us.org/stuff/FX_Phenoelit_25c3_Cisco_IOS.pdf

[11] M. Lynn, "The Holy Grail: Cisco IOS shellcode and exploitation techniques", Black Hat USA, July 2005.

[12] D Pym, B Monahan, "A Structural and Stochastic Modelling Philosophy for Systems Integrity", *HP Labs Technical Report*, 2006.

[13] J. Radianti, Finn Olav Sveen, J.J. Gonzalez, "Assessing Risks of Policies to Patch Software Vulnerabilities", *Proc. Of International System Dynamics Conference,* July 2006.

[14] B Schneier, "Managed Security Monitoring: Closing the Window of Exposure", *Counterpane Internet Security*.

[15] *Symantec Global Internet Security Threat Report: Trends for July–December 07,* Volume XII, April 2008.

[16] G. Zhang, Y. Tan, D. Dey, "Optimal Policies for Security Patch Management", under review.

[17] S. Hariri, G. Qu, T. Dharmagadda, M. Ramkishore, C. S. Raghavendra, "Impact Analysis of Faults and Attacks in Large-Scale Networks", *IEEE Security & Privacy Magazine*, 2003.

[18] Bob Blakley, "Managing Non-quantifiable Security Risks", Burton Group Report, January 2007.

[19] Secunia Advisories by Vendor, http://secunia.com/advisories/

[20] InfoWorld News, "Cisco warns of new hacking toolkit", March 2004, http://www.infoworld.com/article/04/03/29/HNhackingtoolkit_1.html

[21] NetworkWorld News, "Can anyone stop the Cisco exploit?", November 2006, http://www.networkworld.com/reviews/2006/091106-cisco-exploit-test-side.html.

Appendix A. Alternative Probability Distribution Shapes.

One of the assumptions we made about the threat environment was the *shape* of the PDF for exploit development time after patch publication. How much difference does this assumption make to our analysis and can we justify the choice? This section looks at some of the alternative distributions that could be used.

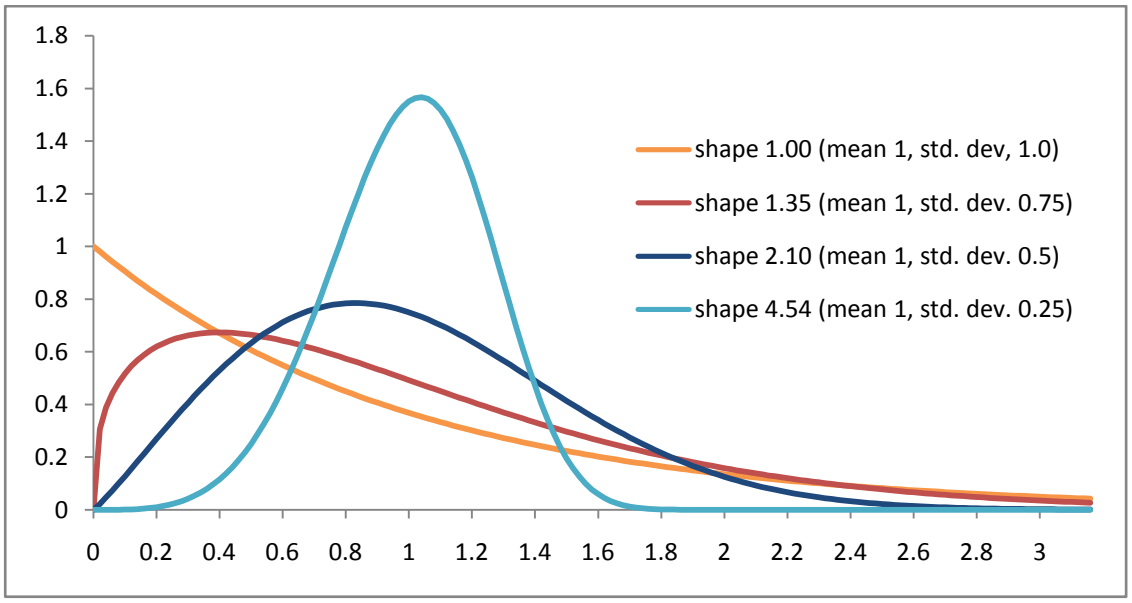


Figure 11. Alternative PDFs of exploit development time

The figure above shows a graph of PDFs of a selection of Weibull distributions with different shape parameters, scaled so they all have a mean of 1.[8] Shape 2.10 is the distribution we chose to use in our current model settings for the emergency arrival. Shape 1.00 corresponds to a "standard" exponential distribution. We rejected it for this model because too high a proportion of exploits arrive a short time after patch publication, and this does not match either the small number of observed cases, or the anecdotal evidence that suggests there are good reasons why exploits take significant time to develop for network devices. Shapes 1.35 and 4.54 illustrate alternative choices we might have made, with greater and lesser standard deviations.

To explore the effect of the distribution shape on the overall disruption, we substituted each of the above distributions for the exploit development time distribution in the model, and ran it over a range of patch deployment policies from 0 to 24 months in 1 month intervals, keeping other parameters of the threat environment, exploit arrival rate and mean development time, fixed at 1 per year and 1 year respectively. The graph below shows how expected emergency disruption varies with deployment policy for each choice of distribution.

We can see that the choice of shape parameter makes a significant difference to the expected level of emergency disruption, especially for deployment policies shorter than 12 months, and this is reflected in the graphs of total disruption below. This is enough to change the optimal choice of patching policy, most obviously for a shape of 4.54.

[8] The ratio of the standard deviation to the mean is determined by the shape parameter.

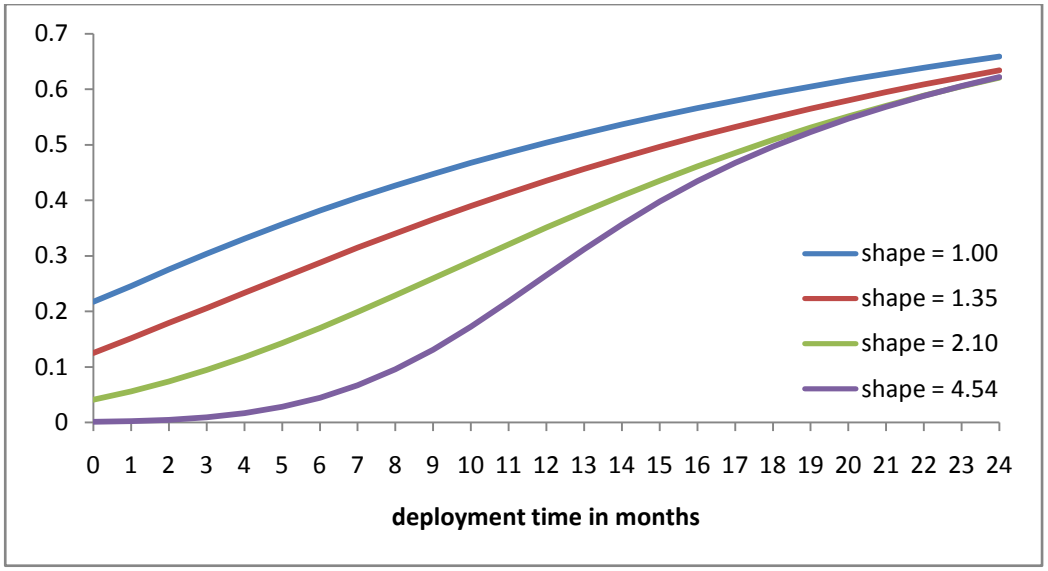


Figure 12. Emergency disruption for different values of shape parameter

One feature in particular, which is clear from the graph of emergency disruption, is that for shape of 1.0 (exponential distribution) and shape of 1.35 there is significant expected emergency disruption even if patch deployment takes zero time. This, however, does not match the history of real network equipment exploits or the experience of network management professionals that we have spoken to, which suggests, at least, that these would not be realistic choices.

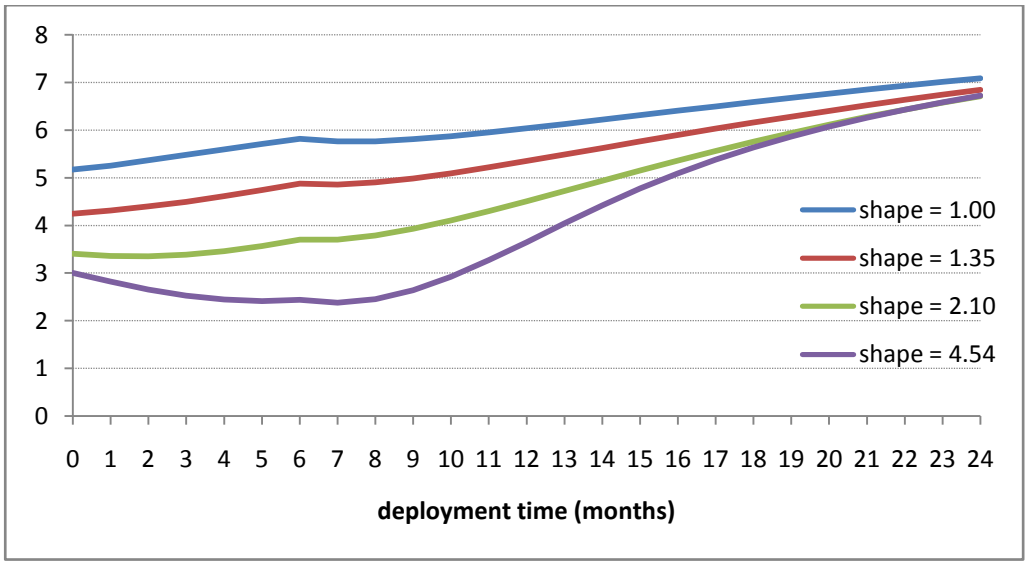


Figure 13. Overall disruption with α=10, for different values of shape parameter

As we don't have enough historical evidence accurately to pin down the shape of the distribution for emergency arrival time, we need an approximation, and the one used in the model seemed to reflect the currently observed threat environment for network equipment. However, the threat environment is ever evolving, and so the appropriate shape might be adjusted in the future. So, as with an emergency arrival rate and mean emergency arrival time, the network security team needs to choose a policy that isn't too brittle with respect to the distribution shape.

Appendix B. Alternative Patch Uptake

Linear uptake of patches across device population is a natural "default" for the current model. It also reflects an ideal target for keeping an even workload on a dedicated network support team, and for keeping all equipment up to date on a regular rolling schedule. From the point of view of the analysis in this paper, it also provides excellent potential for reductions in operational disruption through batching of patches.

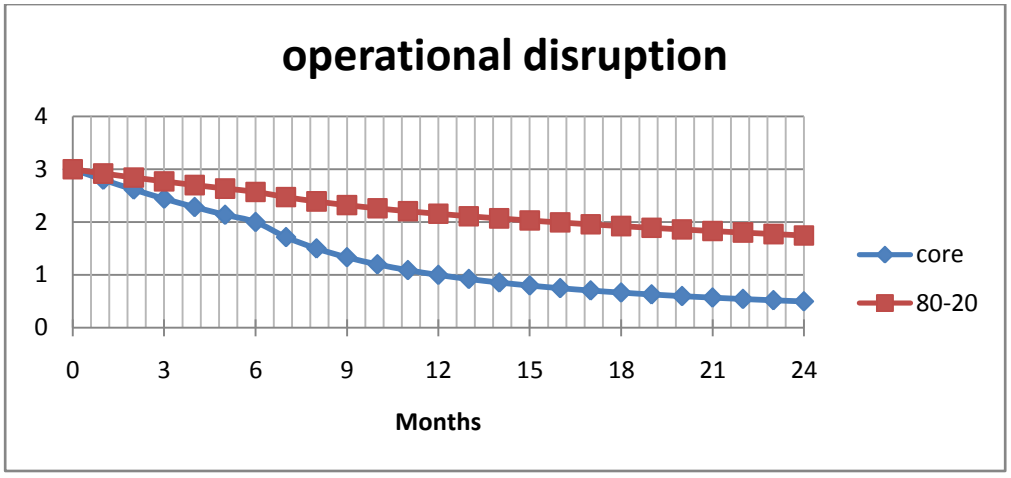
However in practice across many organizations the rate and pattern of patch deployment won't be exactly linear and time to patch is certainly is not distributed evenly across the population of network devices, as network administrators usually choose the best time to patch based on their workload and allowed system downtime schedule, but within the boundaries set by the security policy. A common pattern of deployment is that a majority of the population gets patched relatively quickly, and then the remainder gets patched gradually with the proportion reaching close to 100% near the policy deadline. We use this as an example to illustrate the effects of changing the uptake curve in the model.

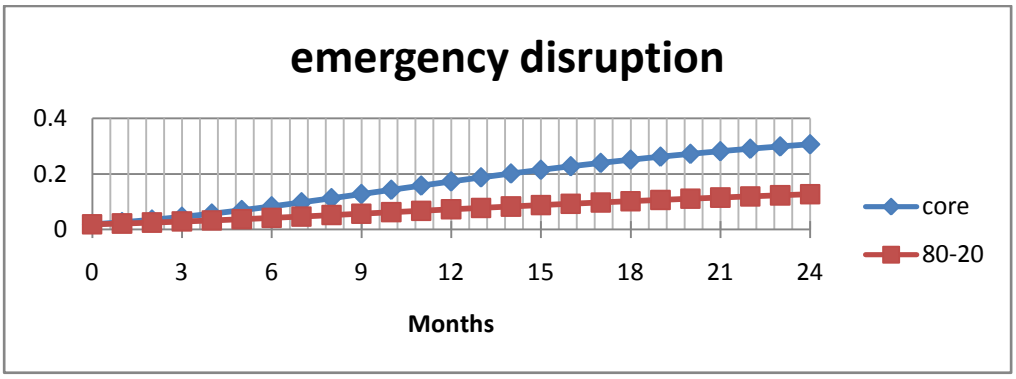
In particular, we have run experiments with a simple "80-20" uptake curve, in which 80% of the vulnerable population gets patched in the first 20% of the allowed deployment time, and the remaining 20% gets patched during the following 80% of the deployment period. It is assumed that if the patch deployment policy is lengthened, the patch uptake continue to follow an 80-20 curve, but taking a proportionally longer time.

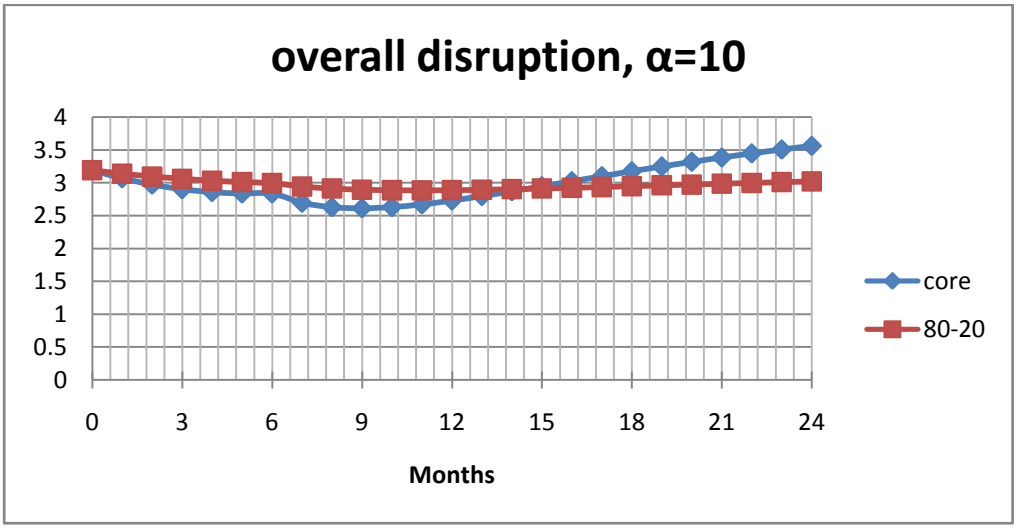
Intuitively, we would expect the new uptake curve to result in smaller reductions in operational disruption, as during the time when 80% of devices are being patched, more quickly than before, there are fewer devices needing other outstanding patches that can be applied in a single batch. At the same time we would expect reduced levels of emergency disruption because of the quicker patch uptake early in the policy period.

Results from running the experiments are shown across the graphs below. We can see that the change in uptake curve makes a noticeable difference in both operational and emergency disruption, with these differences increasing as the patch deployment deadline increases. The graph comparing overall disruption, with $\alpha=10$, shows that there is a significant difference here too, and in fact the change in uptake curve can affect the optimal choice of deployment policy, e.g., for $\alpha=10$ above, it changes from 9 months to 11 months. However, the difference in overall disruption between policies of 9 and 11 months is quite small (2.89 v. 2.88).

Overall, with an 80-20 uptake curve, and with $\alpha=10$, changing the deployment policy has much less effect on expected levels of overall disruption than with the linear uptake. Depending on other conditions, this could be seen as an advantage, e.g., moving to a more relaxed deployment deadline may be beneficial in terms of staff workload, and in terms of effect on total disruption it is a relatively safe move.


operational disruption
4
3
2
1
0
0
3
6
9
12
15
18
21
24
Months
core
80-20


emergency disruption
0.4
0.2
0
0
3
6
9
12
15
18
21
24
Months
core
80-20

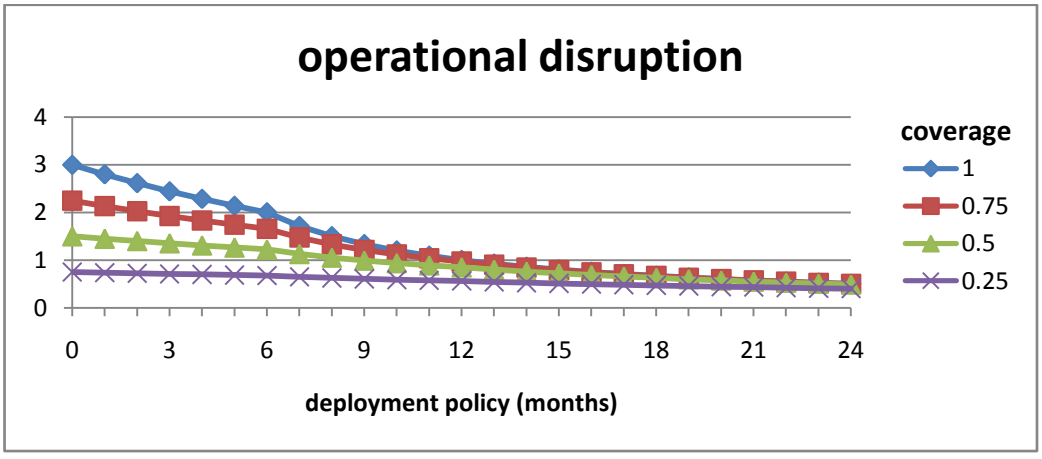

overall disruption, α=10
4
3.5
3
2.5
2
1.5
1
0.5
0
0
3
6
9
12
15
18
21
24
Months
core
80-20

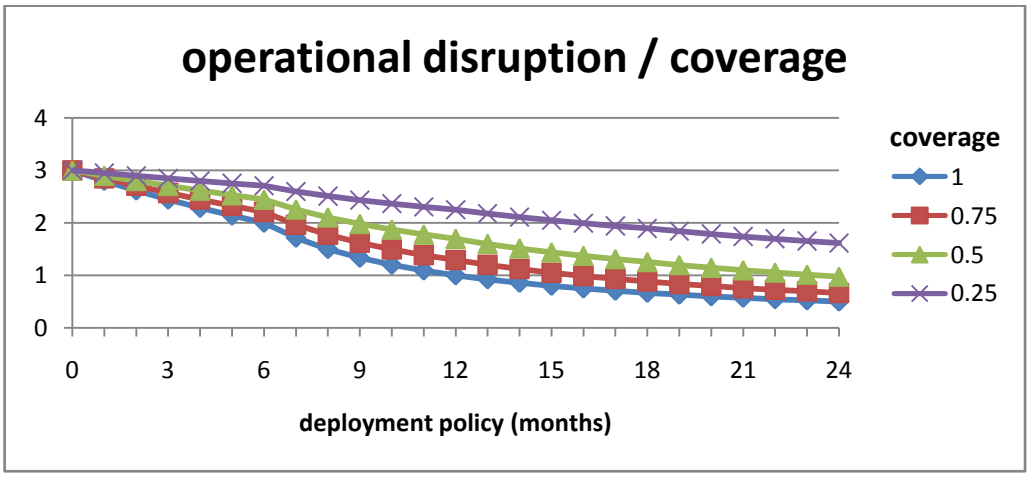
Appendix C. Patch Coverage

In this section we investigate the effects of "patch coverage": the proportion of the population of devices that must be patched each time a patch is released. For our main analysis, we assumed a coverage of 100%, however the alternative would be to select the patch coverage from the interval (0, 1], or sample from a distribution over this interval.

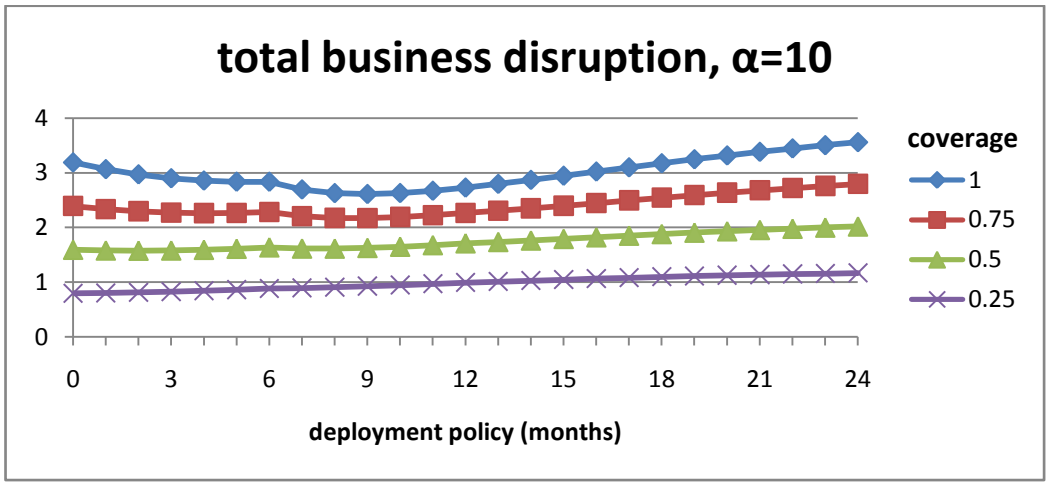
When patch releases are done half-yearly, rather than on a more frequent but *ad hoc* schedule, it is reasonable to assume that a large proportion of devices will need to be patched each time (Cisco has adopted this policy, but not for long enough to confirm this). However other network equipment vendors do not have this policy, so it is important that we explore alternatives, including different expected coverage levels.

We designed a set of experiments to explore the effect of varying patch coverage. We kept the threat environment parameters fixed at 1 emergency per 2 years, with a mean exploit development of 1 year from patch publication, and executed the model with mean patch coverage of 1, 0.75, 0.5 and 0.25. For coverage of 0.75, 0.5 and 0.25 we experimented with fixed values for coverage and uniform distributions centered on these mean values. The results showed that changes in the mean coverage make a significant difference to the overall disruption, and to the resulting choice of deployment timeline, while changes in the standard deviation of coverage makes very little difference (<1% to overall disruption for values of α between 1 and 30). Thus, the graphs below concentrate on changes in the mean value of coverage.





The figures above show how operational disruption varies with deployment policy for different values of patch coverage, first absolute values of disruption, and then disruption divided by coverage, to show the relative effects of changes in coverage. We can see that operational disruption is reduced, but not in proportion to patch coverage, an effect which becomes more marked as patch deployment time increases. (Intuitively this can be explained as, for example, if only half of devices require one patch, and only half require another, then only one quarter of devices require both patches, reducing the relative gain from applying patches in batches.) Emergency disruption is simply reduced in proportion to the mean patch coverage, so we do not show the graphs here.



The impact of coverage on operational disruption is reflected in the graph of overall disruption above, plotted here for α=10, which results in different choices of policy to minimize overall disruption, as follows:

patch coverage	1	0.75	0.5	0.25
deployment policy in months	9	9	2	0

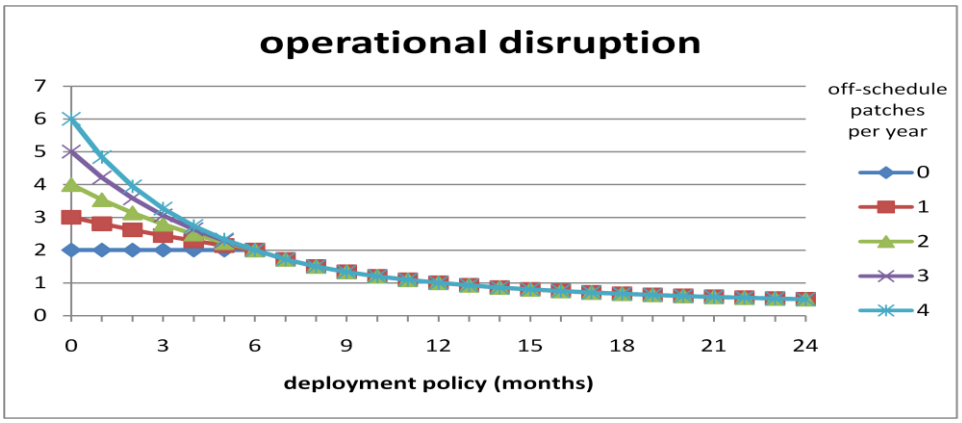
In conclusion, these experiments demonstrate that it is worth having a good estimate of the expected value of patch coverage, as it may affect the best choice of deployment policy. If the patch coverage is generally low, the reductions in operational disruption from delaying patching will not be so great, and quicker patching policies may be better. For the 6-monthly patch release schedule, we think it is reasonable to assume the coverage will be close to 1, and we have used this for our main analysis, lacking detailed data to make a more precise estimate.

Appendix D. Variations in the patch publication rate and release schedule

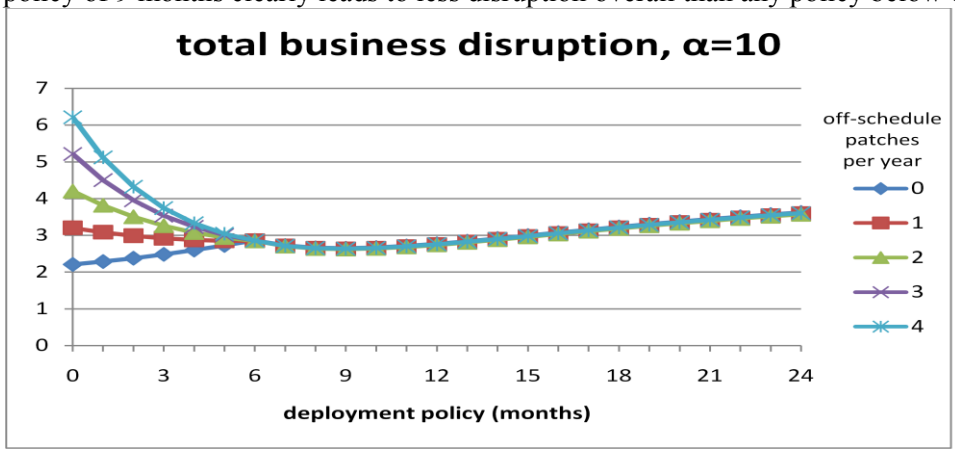
In this section we investigate the effects of different patch release schedules (currently it was set at every 6 months), and, in particular, changes in frequency of off-schedule patches. Since emergency disruption is not affected by the rate of off-schedule patches (as the rate of emergencies is controlled by an independent parameter, and doesn't increase automatically with the number of patches published), we will explore the effects on overall disruption of the patch publication rate, while the threat environment remains the same.

For deployment deadlines of less than 6 months, operational disruption increases when there are more off-schedule patches, because with these short deployment times there is less opportunity for applying patches in batches. When deployment policy reaches 6 months, the curves for operational disruption converge, because with the two regularly scheduled patch publications per year, there is now at least one regularly scheduled patch undergoing deployment at any point during the year, and any off-schedule patches can be batched with one of these.

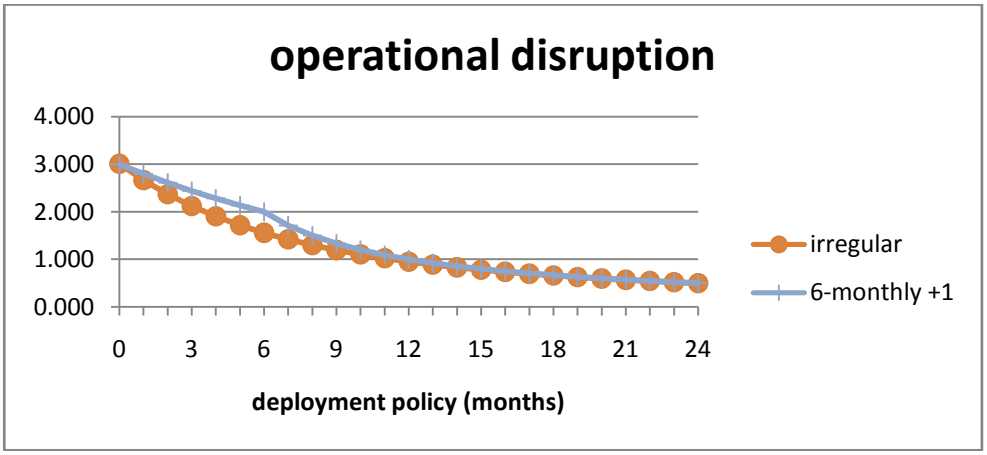
We can also see that when there are no off-schedule patches at all, for deployment policies below 6 months operational disruption remains constant, as each patch is completely deployed before the next one arrives and there is no opportunity for batching.

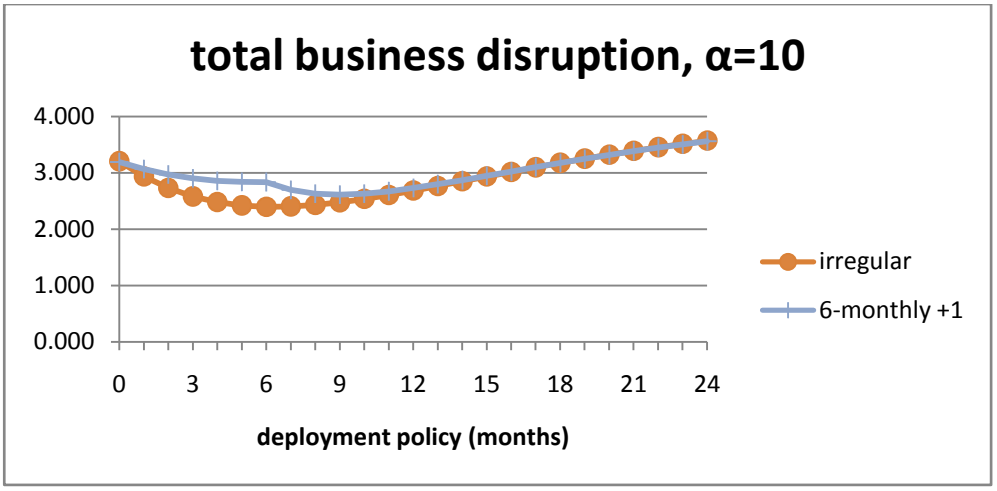


When we look at the graph of overall business disruption, with α=10, we can see that as expected, it is not affected by the off-schedule patch rate for policies of 6 months or greater, but is significantly different for policies below 6 months, to the extent that if there are no off-schedule patches at all the best policy seems to be to patch as quickly as possible, and if there are 2 or more off-schedule patches per year, a deployment policy of 9 months clearly leads to less disruption overall than any policy below 6 months.



In the graphs above we assumed a regular schedule of 2 patch publications per year, with some rate of publication of further patches at irregular intervals. We also investigate what happens if we keep the same average rate of patch publication, but without the regular schedule. For this experiment, we ran the model with the time interval between patch publications sampled from an exponential distribution, mean 1/3 year, i.e., on average 3 patches per year as before.





Looking at the graphs of operational disruption and overall business disruption, we can see that the regular schedule makes a significant difference to expected levels of operational disruption. (As before, expected emergency disruption is not affected by the patch publication schedule.) This shows up particularly for deployment policies in the region of 6 months, and without the regularly schedule the deployment policy which minimizes business disruption is 6 months rather than 9 months (which suggests that the regular patch publication schedule was a detail worth including in the model).